THE EFFECTS OF TOOL CONTAINER LOCATION ON USER PERFORMANCE IN GRAPHICAL USER INTERFACES

A Thesis Submitted to the College of

Graduate Studies and Research

In Partial Fulfillment of the Requirements

For the Degree of Master of Science

In the Department of Computer Science

University of Saskatchewan

Saskatoon, CANADA

By

André Doucette

PERMISSION TO USE

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

ABSTRACT

A common way of organizing Windows, Icons, Menus, and Pointers (WIMP) interfaces is to group tools into tool containers, providing one visual representation. Common tool containers include toolbars and menus, as well as more complex tool containers, like Microsoft Office's Ribbon, Toolglasses, and marking menus. The location of tool containers has been studied extensively in the past using Fitts's Law, which governs selection time; however, selection time is only one aspect of user performance. In this thesis, I show that tool container location affects other aspects of user performance, specifically attention and awareness. The problem investigated in this thesis is that designers lack an understanding of the effects of tool container location on two important user performance factors: attention and group awareness. My solution is to provide an initial understanding of the effects of tool container location on these factors. In solving this problem, I developed a taxonomy of tool container location, and carried out two research studies. The two research studies investigated tool container location in two contexts: single-user performance with desktop interfaces, and group performance in tabletop interfaces. Through the two studies, I was able to show that tool container location does affect attention and group awareness, and to provide new recommendations for interface designers.

ACKNOWLEDGMENTS

I would like to thank my co-supervisors, Carl Gutwin and Regan Mandryk, for always lending an ear and guiding me through this process; this thesis would not have been the same without them. I'd also like to thank all the members of the Interaction Lab; you made it a fantastic work environment. Lastly, I'd like to thank my parents, Nick and Leona, for always encouraging me to continue, and Stephanie, who has always been there and to whom I owe so much.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS/ACRONYMS

CAD	Computer Aided Design
DPI	Dots per inch
GUI	Graphical User Interface
HCI	Human Computer Interaction
IM	Instant messaging
IPI	In-place Interaction
IRC	Internet relay chat
px	Pixel
RT	Response time
SC	Shadow Cursor
ST	Static Toolbar
VT	Visual Trackpad
WC	Warp Cursor
WIMP	Windows, Icons, Menus, and Pointers

CHAPTER 1

INTRODUCTION

Although first developed nearly 30 years ago, most consumer computer interfaces are still based on the Windows, Icons, Menus, and Pointer (WIMP) interface approach of the Xerox Star [34]. There are many reasons why WIMP interfaces have become so common, including being easier to learn than text-based interfaces and offering direct manipulation tools [48].

A WIMP interface is made of multiple windows, each containing icons and menus specific to that application. Each of the items represented in the menus or by the icons is known as a graphical user interface (GUI) tool, and these GUI tools are how users are able to interact with the application. Most work in WIMP interfaces is done through the use of these GUI tools. Common examples of GUI tools include the bolding tool in text editors, the paint fill tool in MSPaint, and the navigation buttons in web browsers (see Figure 1). GUI tools must have a visual representation, and often have a label (Figure 2a), an icon (Figure 2b), or both (Figure 2c).

Figure 1 - a) bolding in MS Word 2007, b) paint fill in MSPaint, c) navigation buttons in Chrome

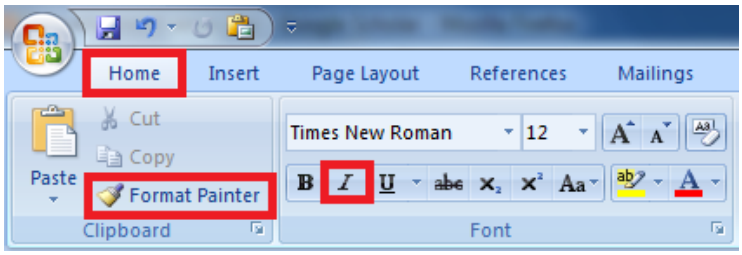


Figure 2 - MS Word 2007 Ribbon: a) home tab, b) italics tool, and c) format painter

To help organize the interface, these tools are often stored in tool containers. There are many examples of tool containers, including toolbars, menus, and variations (e.g., Microsoft's Ribbon).

A tool container groups multiple GUI tools into one visual representation and often allows tool activation via simple selections (e.g., a mouse click); for example, common examples of tool containers are toolbars and hierarchical menus. Other more complex tool containers are common in the HCI literature, including gestural interfaces, such as Marking Menus [42], and bimanual techniques, such as Toolglasses [10].

Regardless of the structure of the tool container, tool containers store GUI tools that can typically be activated via a simple selection. In common computer environments, this selection is a mouse click, or a tap of a finger or stylus on a touch-sensitive display. To distinguish between other selections that occur in the user's workspace, I call this interaction a "tool selection".

Since they have a visual representation, tool containers must be displayed on-screen. Common computing environments provide users with relatively large workspaces. Most users have large displays (at least 1024x768), and multi-display environments are becoming common. To make use of this large workspace, current applications often display tool containers along the edges of the screen (Figure 3a), attempting to maximize the area available for the user's work artifacts. Other applications offer moveable floating containers (Figure 3b) that allow users to define the location of the containers. These allow users to organize their workspace to best support their work, including moving these tool containers to a secondary display. A third common tool container location is a user-defined popup location (Figure 3c). Popup tool containers conserve the most workspace real estate, because they are hidden from view until needed. They, however, require a mode switch before tool selections can be made.

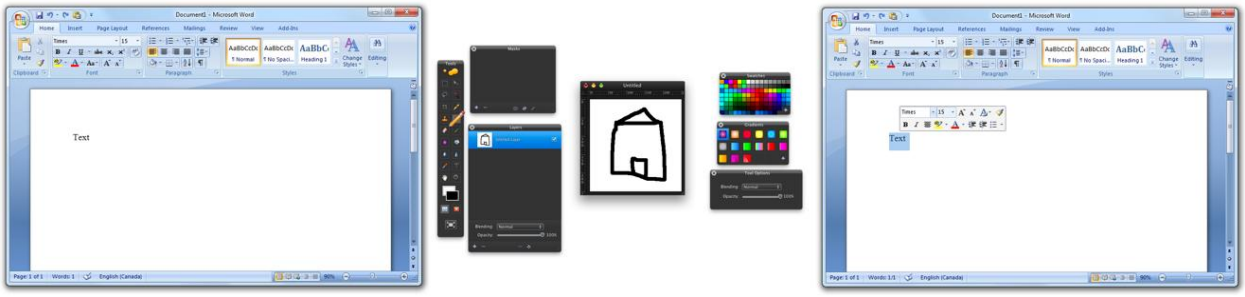


Figure 3 - a) MSWord Ribbon, b) PixelMator floating palettes, and c) MSWord 2007 popup tool

The choice of location for a tool container affects a user's performance in selecting tools from that container, and when tool selections occur several times during a task, location may have a substantial effect on their overall task performance. One well known effect of tool container location is the effect that distance has on tool selection time. For example, researchers have used Fitts's Law (the proportional relationship between the distance to a target and its width to the movement time required to acquire the target) to model tool selection time in GUI applications. Movement time, however, is not the only consideration of performance, and previous research has not considered other factors affected by location.

One example of the effect of tool container location is that of attention. When working in computer applications, users have a point of focus. This is typically their point of interaction: cursor, stylus, or finger. Tool selections that are outside of this point of focus require users to shift their attention away from their work, then make a selection, and finally return to their workspace. Designers still do not understand the attentional costs of displaying tool containers at different locations outside of this point of focus. For example, would it be better to have a nearby tool container on a secondary display, or a more distant tool container on the same display? In Figure 4, the user wishes to change the brush size while drawing the house's door (bottom left of main display). In Figure 4a, the brushes palette is far on the same display, and in Figure 4b, the brushes palette is close, but on another display.

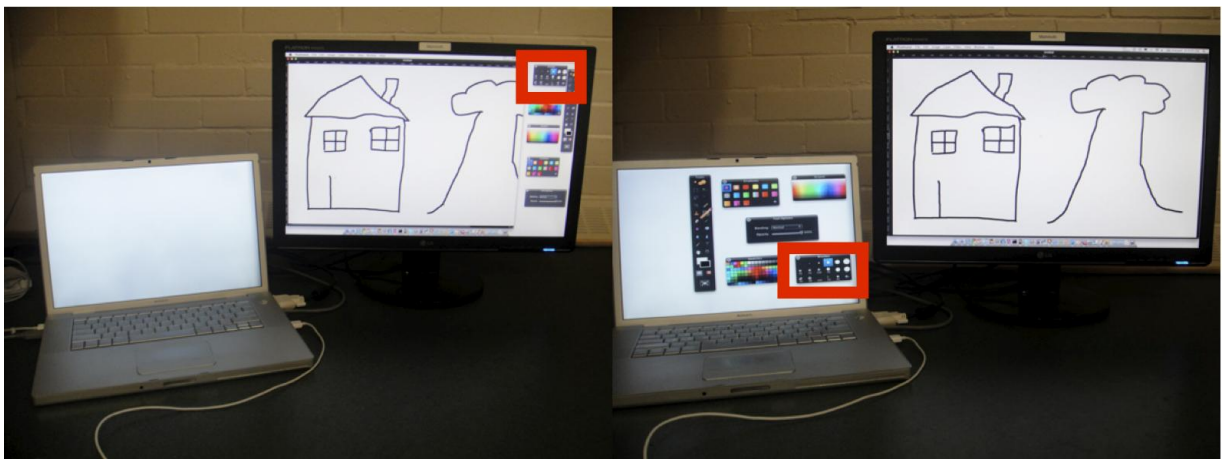


Figure 4 - Change brush size while drawing house's door. The brushes palette is:
a) far on same display, b) near on another display

A second example of how tool container location affects performance involves situations where a group of people works together in a shared interface. One important aspect of effective group work is group awareness; interfaces that do not provide good awareness do not effectively support group work [32]. In the real-world, tables are often used for group work, and locations on the table have conceptual meanings. For example, tools in the center of the table are thought of as "our tools" and tools in each of the users' workspaces can be thought of as "their tools". We still do not, however, fully understand the effect that displaying tools in these locations has on the ability for the group to accomplish coordinated work. For example, although moving the tools closer to each user makes selections easier, it is not clear whether the other users are still able to maintain awareness when tool selections occur in users' personal workspaces.

These two examples illustrate two important aspects of user performance that stem from tool container location, but about which we have little understanding: attention and group awareness.

1.1 PROBLEM

The problem investigated in this thesis is that there is a lack of empirical evidence showing how container location affects user performance – particularly, attentional costs and group awareness.

1.2 MOTIVATION

It is important to build a better understanding of how tool container location interacts with other performance factors such as attention and awareness because without this knowledge, GUIs cause errors, confusion, and breakdowns in coordination between users.

Attention

Interactions that occur at a user's point of focus incur little additional attentional costs, because users are already focusing at that location. Tool containers that are displayed outside of this point

of focus require a visual shift from the workspace to the container and back again. This is a costly operation, because it may break a user's task flow, causing them to repeat work that had already been accomplished in order to resume working. Also, the visual shift to the tool container requires additional visual search time, as users must visually acquire the tool container displayed at some location out of their point of interaction. In this thesis, I investigate these two attentional costs: the cost of shifting visual attention, and the cost of visual searches.

There are other potential attentional context switches - other than visual shifts - caused by container location that will affect user performance. For example, displaying a tool container on an external touchscreen requires a large visual shift, but also requires users to switch from regular indirect mouse input to direct touchscreen input, and back again. This attention switch between using two different input types can cause mode errors, as users may be confused about which mode they are in.

There are already consumer technologies in existence that make these problems real. For example, external touchscreen monitors (http://www.mimomonitors.com/) and touchscreen laptop trackpads (http://www.sharp.co.jp/mebius/products/pcnj70a/index.html) both afford a touchscreen that could easily house tool containers. Because these technologies already exist, it is important that we understand how these additional context switches affect user performance before widespread implementation because users may be confused by their use.

Thus, not understanding how container location affects attention and performance could dramatically affect a user's tool selection performance, requiring more time and effort to undo errors, and causing users to become frustrated.

Group Awareness

The tools designers provide to users must support both individual work and group work; thus, there is a tradeoff between providing powerful tools for individuals and powerful tools for the group. Although we know awareness is important for effective collaborative work, we still do not fully understand which factors affect group awareness.

One factor that has not been looked at is that of tool container location. For example, in collocated tabletop applications, we know by Fitts's Law that displaying a tool container in-place, at the user's interaction point, is the optimal location for individuals. With these in-place tool containers, however, users may find it difficult to stay aware of other users' actions. The lack of group awareness makes effective collaboration difficult, causing collisions and repeated work, and requiring additional time and effort to coordinate the group work.

Thus, not understanding how container location affects awareness could dramatically affect a group's ability to stay aware of each other's activities, which can cause collisions and repeated work, and so hinders their ability to work together smoothly and naturally.

1.3 SOLUTION

My solution is to provide an initial understanding of how tool location affects attention and awareness, through empirical and theoretical evaluations. In the individual case, I want to show the effect of tool container location on attention and tool selection performance. In the group case, I want to show how container location affects awareness, demonstrating the tradeoff between providing powerful tools for the individual and powerful tools for the group.

1.4 STEPS TO THE SOLUTION

Three main steps were carried out for this solution: a taxonomy of tool container location, a study of how location affects attention, and a study of how location affects group awareness.

Develop a taxonomy of location

The full set of possible tool container locations is very large, so it was necessary to group locations into different categories to reduce the number of possibilities. I thus developed a taxonomy of location, described in Chapter Three. This taxonomy organizes tool selection

techniques by their visual representation location, and outlines performance issues that arise from these locations. I used this taxonomy to describe three conceptual locations (in-place, inside workspace, and outside workspace) which I then used to create the techniques studied in Chapter Four and Chapter Five.

Study the effects of container location on attention

The first study looked at the interaction between container location and individual attention. I looked in particular at in-place techniques and their access method. I developed three in-place techniques, one in each of the conceptual locations from the taxonomy, and compared their performance in an experimental study.

I found that there are substantial effects of tool container location on individual attention.

Study the effects of container location on group awareness

The second study looked at the effects of container location on group awareness. I looked in particular at awareness in a tightly-coupled task, and the effect of the demands of the individual on group awareness.

First, I developed a technique for each of the three conceptual locations in the taxonomy. Next, I compared group awareness costs among the three tool container locations, as well as among differing individual task demands, in an experimental study. From the results of the study, I showed that container location does affect group awareness, and confirm that public locations offer better awareness than personal workspaces.

Provide design recommendations

From my two studies, I created a set of recommendations for designers of GUI applications. I provide recommendations for two contexts: individual work and group work. These recommendations allow designers to create more powerful tools for individual work, and also tools that can effectively support group work.

1.5 CONTRIBUTIONS

The main contribution of this thesis is the new understanding of how tool container affects attention and group awareness, as described in Chapter Four and Chapter Five. From these studies, I created design recommendations for individual and group contexts, providing actionable recommendations in the design of tool containers.

There are several secondary contributions of this thesis. First, I developed a taxonomy of location (Chapter Three) that was used to put my work on tool container location in context. Second, I compared three in-place toolbar access techniques (Chapter Four), and provide empirical results of the benefits and drawbacks to the different access techniques. Third, I developed and tested a novel input device and technique (Visual Trackpad, described in Chapter Four). Fourth, I provide an experimental environment where researchers can investigate effects of tool container properties on group awareness (presented in Chapter Five).

1.6 THESIS OUTLINE

In Chapter Two, I outline the related research and products which will form the foundation I build upon in this thesis. I begin with an overview of tool containers and selection techniques, a discussion of how location has been studied in the past, and finish with overviews of attention and group awareness.

Chapter Three introduces a taxonomy of location, which divides the design space for tool containers into three conceptual locations, and provides context for the investigation of the effects of tool container locations on user performance.

Chapter Four presents an evaluation of three toolbars, displayed in each of the taxonomy locations, and presents empirical results showing the effects of toolbar location on attention.

Chapter Five presents an empirical study of three toolbar locations, and presents empirical results showing the effects of toolbar location on group awareness.

Chapter Six discusses the results from the two studies, and outlines the design recommendations that stem from them.

Finally, Chapter Seven summarizes the work completed in this thesis, the contributions it makes, and work that should follow on from it.

CHAPTER 2

RELATED WORK

This thesis is concerned with providing additional understanding of the effects of tool container location on individual attention and group awareness. To this end, there are four areas of interest in the research literature. First, it is important to place the discussion of tool containers in context by providing an overview of existing tool containers and selection techniques. I show in this thesis that the location of a tool container is an important factor that must be considered when developing GUI applications. Thus, the second section describes how location has been investigated in the past, providing context for the taxonomy in Chapter Three. Third, I provide an overview of attention, and how it has been investigated in HCI. Similarly, an overview of awareness, including the types and how it has been investigated in the past, is presented in the fourth section.

2.1 CONTAINERS AND SELECTION TECHNIQUES

In GUI applications, *selection* is often referred to as the combination of *pointing* and *target acquisition*. In most applications, users indicate which item they want to select by first pointing at the item and then invoking the selection (often, this is a mouse click). A selection can invoke a command (*command selection* [23,29,36,50]) or can pick a tool to interact in the workspace (*tool selection* [48]). Each item that can be selected, whether it is a command or a tool, is called a *GUI tool*. GUI tools have visual representations and are accessed with a simple selection.

Often, multiple GUI tools are grouped into tool containers. A container always has an associated GUI tool selection technique. This is often a simple mouse click, but there are other more complex selection techniques, including bimanual interactions [10,43] and gestures [42].

2.1.1 Containers

A tool container is a GUI artifact that groups multiple GUI tools into one visual representation. There are many examples of containers in the literature [4,10,12,22,23,29,36,43,54,56,64]. These can be grouped by aspects of their visual representations: static visual representations, movable containers, and hidden (popup) containers.

Static visual representation

The first group of containers is those with static visual representations. These containers have a visual representation on screen at all times, and their representation location is fixed. Two common examples of static visual representation containers are static toolbars [15,23,50] and linear menus [4,12]. These containers are common because the list of possible commands is visible and thus are easy to use and easy to learn for beginners [26]. Most GUI applications have one or more toolbars in their interface (for example, see Figure 5a). Also, current operating systems (e.g., Windows, Mac, Linux) have menus associated with their applications (displayed in the application window or at the top of the screen – see Figure 5b).

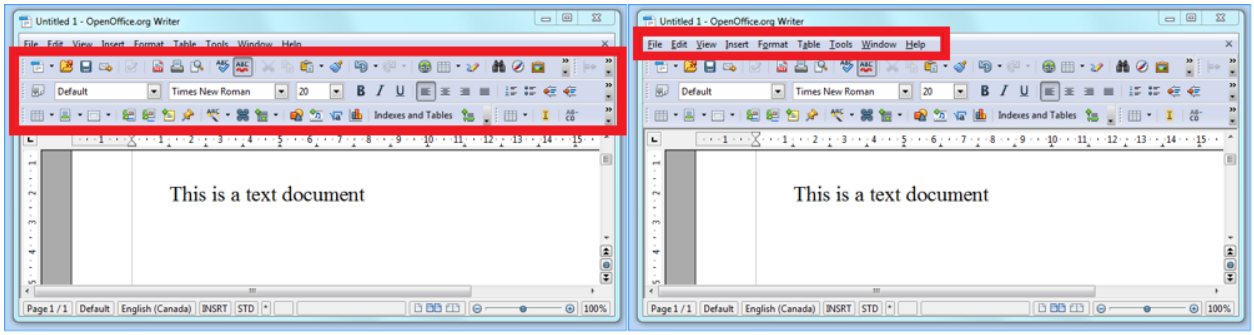


Figure 5 - OpenOffice; a) stacked toolbars, b) menus along top of window

Although toolbars and menus are common, they are slower than other more advanced techniques, such as keyboard shortcuts [44]. One reason they are slower is that they are often displayed along the edges of the screen making them the farthest targets to reach; however, all standard menus are not created equally. A menu displayed along the edge of the screen will be faster to select from because it has essentially an infinite target width in motor space [1].

There have been several attempts to improve static tool containers. For example, split menus [62] offer recency- or frequency-based layout of items, and have been shown to be faster than regular menus [62]. Even simple toolbars have been modified. Microsoft Office 2007 (http://office.microsoft.com) has introduced a new toolbar called the Ribbon. The Ribbon is a tabbed based, contextual toolbar, and replaces the previous menu system.

The LensMouse [71] is a touchscreen attached to the top of a regular mouse (see Figure 6). The authors in [71] tested LensMouse as a direct-touch tool palette. In their study, the tool palette displayed was static, making LensMouse a static visual representation container; however, this technique could easily be used for dynamic containers: contextual tools, for example, that change depending on which item is selected in the workspace.

Figure 6 - LensMouse prototype (from [71])

Moveable containers

The second type of tool container is those that are moveable. Moveable containers have a dynamic display location. They can be moved closer to a user's workspace, making them faster to use [47]. Moveable containers also allow users to define the location they want to display the container, allowing users to organize their workspace to best support their work.

A common moveable container is a floating palette. These are common in CAD, drawing, and image manipulation applications because users only need access to a small portion of the available commands at a time. They are also used in other applications, such as contextual Inspector windows in OS X applications (see Figure 7).

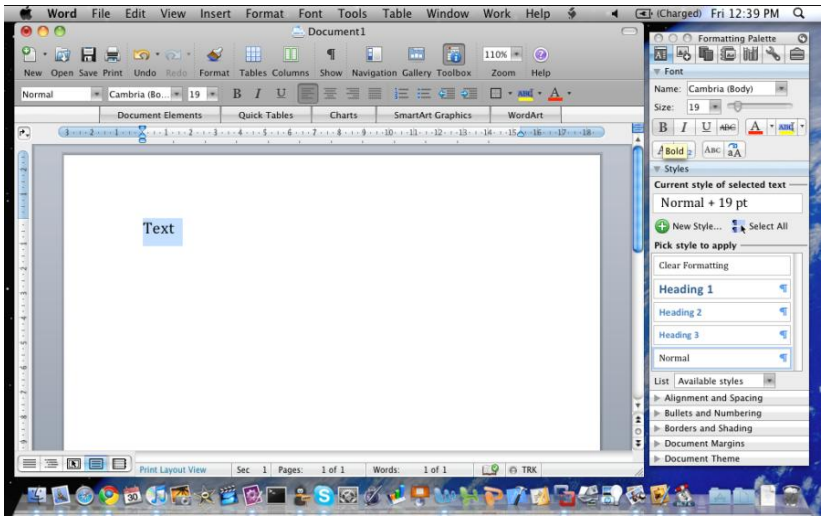


Figure 7 - Inspector window on right side of MSWord for Mac

Another type of moveable container is one that is controlled by another input device. For example, Toolglass [10] is a bimanual interaction technique that makes use of a second input device operated by a user's non-dominant hand. A Toolglass contains sheets of GUI tools that are selected using the user's dominant hand. The sheets are transparent, offering a type of interaction called a see-through interface [10] (see Figure 8). Studies of Toolglasses have shown they are efficient [50].

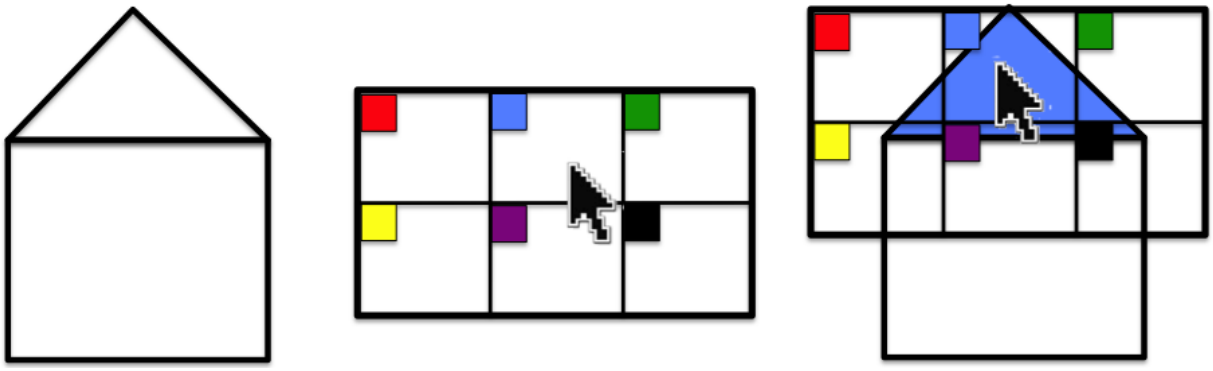


Figure 8 – Example of Toolglass sheet; a) the workspace item (a house), b) fill-tool sheet moved by the user's non-dominant hand, c) click through blue color with the cursor moved by the user's dominant hand, modifying workspace item below the sheet

The last set of moveable containers is those that are not explicitly moved by the user. Instead, these containers follow the point of interaction (the cursor or stylus) so they are always close-by.

The main motivation for this type of tool is to reduce "tool palette round trips", as described by Fitzmaurize et al. [22], which are monotonous when switching frequently between tools.

An example of following containers are Tracking menus [22], menus displayed above the user's point of interaction. Users can move within the menu without moving the container, but when the stylus makes contact with the edges of the menu, the container moves with it. Fitzmaurize et al. [22] found that tracking menus were good for panning and scrolling in large documents.

Another following tool switching technique is PieCursor [23]. This moveable container is based on Tracking menus in that it displays a radial menu above the cursor; however, in PieCursor, the movement of the cursor highlights one of the options in the menu. A click then selects that menu item, and dragging performs that operation (for example, zooming or panning).

Popup containers

The last group of containers is popup containers. These containers are hidden from view, and when a modifier key or gesture is performed, the container is displayed. There are two types of popup containers: those that pop up at a fixed screen location and those that pop up above the current point of interaction.

Containers that pop up at a fixed screen location

The Springboard [36] is a popup menu on tablet PCs that is displayed in the bottom left corner of the tablet. Its location is constant, as it is always displayed in the tool lagoon.

Visual Trackpad [15] is a direct touch popup toolbar developed for study 1, and presented in Chapter Four. A trackpad is used as a regular mouse, and a mode switch displays a toolbar on the trackpad's touchscreen, allowing for direct input toolbar selections. Visual Trackpad displays the toolbar on another device (i.e., not the main screen), and thus does not pop up above the cursor. The LensMouse [71] is implemented similarly, but does not use a trackpad for indirect input; instead, the physical movement of LensMouse provides indirect input, as with a regular mouse.

These examples of popup containers appear at the same screen location each time they are activated. They can appear at the same location for consistency, such as the Springboard [36], or because of a physical restriction, such as Visual Trackpad [15] and LensMouse [71].

Containers that pop up around the point of interaction

The next group of tool containers are those that pop up above the user's point of interaction (either above the cursor, or around the stylus). The most common popup tool containers are context menus. In current operating systems, these are typically displayed after a right click occurs. A context menu contains commands that are appropriate for the item clicked, and selected commands only modify the item that was clicked.

Above-cursor popup toolbars also exist. For example, in Microsoft Word 2007, highlighting a piece of text and performing a short "up and right" gesture displays a toolbar just above the text (see Figure 9). This toolbar contains commonly used text commands, such as bold, fonts, and bulleting. Another example of a popup toolbar that appears above the user's cursor is Shadow Cursor [15]. Once a modifier key is depressed, a toolbar appears above the cursor, replacing the system cursor with a shadow cursor. Users control the shadow cursor and make a selection from the toolbar, which hides the toolbar and replaces the system cursor. The *clickaround* tool-based graphical interface [9] is another example of an above-cursor popup toolbar, but uses two mice and two cursors.

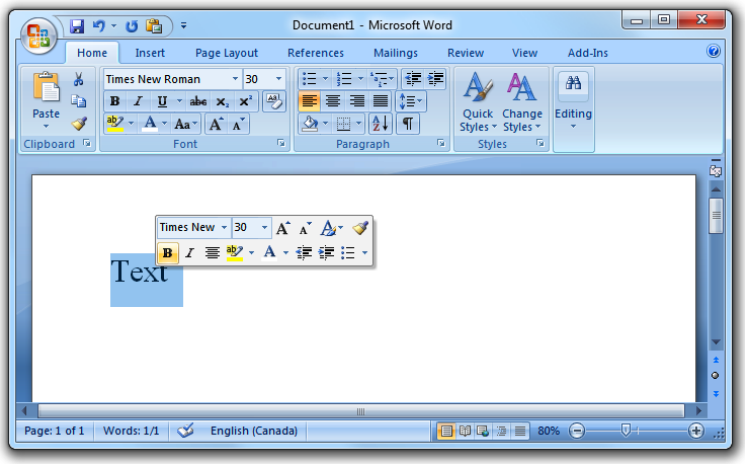


Figure 9 - MSWord 2007 popup toolbar

Pie menus [12] (see Figure 10) and a modification of them, floating pie menus [56], are another type of above-cursor popup tool container. Although pie menus could also be implemented as static visual representation containers [12], typical implementations of pie (or radial) menus are above-cursor popups. These menus arrange the menu items in a circle around the cursor instead of in a linear list. This arrangement allows each element to be selected with the same effort, but in different directions (as opposed to linear menus, where the access time is proportional to the number of items in the menu [12]).

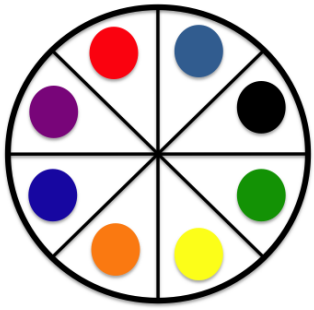


Figure 10 - Example of an advanced color-selection pie menu

The Hotbox [43] technique is an above-cursor popup tool container. It contains many different types of containers within it, including modal dialogs, menus, radial menus, and marking menus. The Hotbox was implemented for the 3D animation and design application Maya (http://www.autodesk.com). Due to the large number of commands in the application (over 1200 [43]), the menubar system previously employed was overloaded and cumbersome. The new design, the Hotbox, enabled quick access to commonly-used commands via a series of marking menus, and made access to other commands still available through various other menubars displayed within it.

Optional visual representation

There are other tools that pop up above the user's cursor, but these tools are operated via gestures. They support expert usage by not displaying the tool container if the gesture occurs fast enough. The tool container can still be displayed if the user pauses for a specified period [29,42,54].

The three tool containers described in this section are based on marking menus [42]. A marking menu is a popup radial menu that appears above the user's point of interaction (in [42], the point of interaction is the location of the stylus). Items from marking menus are selected by making a mark in the direction of the tool to select in the radial menu. Marking menus can be linear or hierarchical [73], and should be no larger than 8 items 2 levels deep or 4 items 4 levels deep.

Two containers based on marking menus are FlowMenus [29] and control menus [54]. A FlowMenu is a hierarchical marking menu that includes command and parameter entry in a single gesture. The first mark selects the command, then following marks select the parameters for the command. For example, when the first mark selects the zoom command, the next mark selects the zoom level [54]. Control menus are similar to FlowMenus, except they include other interface elements to select the parameters. Taking the zoom example from above, a control menu may display a scroll bar that can change the zoom level, instead of requiring users to enter a numerical value, as with FlowMenus.

In conclusion, a tool container has two main roles. The first is to display the available options that can be selected. The second is to allow users to make selections from them. Next, I continue the discussion of tool containers by describing different selection techniques employed by example containers from the literature.

2.1.2 Selection techniques

The main purpose of a tool container is to provide a way for users to make command and tool selections from the available items. Users select items from tool containers using a selection technique. Selection techniques can be grouped by their selection mechanism: simple selection or gestures.

Simple selections

The overwhelming majority of selection techniques employed in current systems are simple selection techniques. These selection techniques can be grouped into three types: cursor clicks, direct touch taps, and the selection of one item in an area.

Cursor clicks

Cursor input is the most prominent spatial input mechanism in current systems. Cursors are generally controlled by input devices such as mice, trackpads, and trackballs. These devices provide relative 2D input, allowing users to position a virtual cursor at any point on a 2D screen. Many of these devices provide multiple selection methods, canonically known as left-clicks and right-clicks. Many tool containers (e.g., [5,10,15,22,23,25,48,56]) activate a tool when users perform a single left-click on an item.

Direct touch taps

Another form of simple selection occurs in direct touch systems, such as touchscreens and tablet PCs. In these systems there is no mouse, so users enter spatial 2D input via their finger or stylus. Simple selections occur via a tap on the screen at the location users want to make a selection. Examples of containers that have this type of selection are [10,15,36,71].

Selection of one item in an area

One limiting factor of typical cursor input or direct touch input is the speed at which a user can move the point of interaction. This speed has been modeled by Fitts's Law, discussed next in Section 2.2.1; however, many researchers have attempted to increase human performance by selecting using areas instead of points [8].

The first example of this type of selection technique is area cursor [39]. Area cursors define a spatial region on the screen that is controlled by the mouse. This area is larger than the single point of regular cursors, and so makes selection of very small targets easier. There is confusion of which item in the area that will be selected, however, when targets are densely packed.

A second technique, Bubble Cursor [27], expands on the idea of area cursors but instead defines the area dynamically. Bubble Cursor dynamically divides the entire screen into selection regions such that one item is in each region at all times. Thus, Bubble Cursor increases the activation area of each target, instead of modifying the activation area of the cursor, as with area cursors.

Gestures

In contrast to simple selection techniques, which either require a mouse click or a direct touch tap, some selection techniques accept more complex input. One complex input type is gestures. Both the Hotbox [43] and marking menu [42] containers accept gestural input. In both cases, the gestures are directional marks that indicate the direction of the item the user wishes to select.

Another type of gestural selection technique is the "finger gestures" on trackpads, Apple's Magic Mouse (http://www.apple.com/magicmouse/), and in Microsoft Research's five prototype multitouch mice [69]. These gestures are not directly tied to a tool container, but some of the gestural selections do select items from tool containers. For example, the 3-finger left swipe on Apple laptops is known as the "back" gesture, and works across multiple applications by invoking the "back" button of that application (whatever "back" means in this context).

2.1.3 High-level models of selection

Fitts's Law, discussed in Section 2.2.1, is a low-level model of selection. There are, however, a few high-level models of selection, including Buxton's three-state model, keystroke-level modeling (KLM), and GOMS (goals, operators, methods, and selection rules), although none of these models considers selection beyond the issue of movement time.

Three-state model

The first high-level model of selection presented is Buxton's three-state model [11]. The full three-state model assumes a hover state, such as that present in stylus-based interaction. A sub-set of the full model is presented in Figure 11, and forms the basis of this discussion.

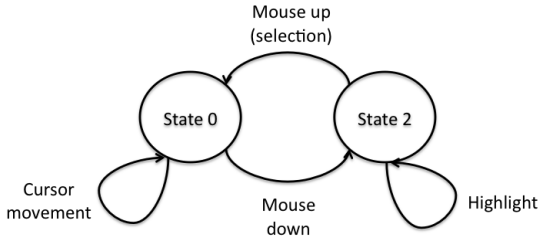


Figure 11 - Buxon's three-state model (simplified, based on [11])

This model was created in order to provide a way to compare different task requirements and match them to correspondingly capable input devices [11]. In Figure 11, state 0 represents regular mouse cursor movement, whereas state 2 represents a click-and-drag action.

Keystroke-level modeling (KLM)

KLM was designed with one goal in mind: "…predicting one aspect of performance: the time it takes an expect user to perform a given task on a given computer system" [13]. The model assumes that a task can be split into sets made up of five operators: K (keystroking), P (pointing), H (homing), D (drawing), and M (mental preparation). The total time required to complete a task, without error, is the sum of the operators required to accomplish this task.

Card et al [13] also provide a series of time estimates for common tasks. For example, a pointing operator takes about 1.1 seconds and a keystroke takes the best typist about 0.08 seconds to complete.

GOMS

A third main model of high-level selection is GOMS – Goals, Operators, Methods, and Selection rules [38]. In this model, goals can be thought of as the task to accomplish - for example, delete a phrase or insert a word. The operators are the actions that need to occur to accomplish the goal – for example, click a mouse button or move the mouse. The methods are the sequences of operators that occur in order to accomplish a goal – for example, click mouse (select beginning of word), move mouse (highlight word), press delete (the three operators together accomplish the goal delete word). Lastly, the selection rules outline which of the methods will be selected to accomplish a goal – for example, you can delete a word by the method outline above, or by clicking at the end of the word and hitting the backspace key multiple times [38].

There are many variants of GOMS, but their discussion is outside the scope of this thesis. GOMS has been used in different application domains in order to model the performance of an expert doing work in those applications, in order to find bottlenecks in the processes offered by the application.

2.2 LOCATION

For this thesis, I am interested in how the choice of location of tool containers affects individual attention and group awareness. In this case, location refers to the physical location of the visual representation of the tool containers, which includes the screen on which it is displayed as well as the location of the tool container on that screen.

Other than the location of the visual representation of a tool container, there is also the physical location of input to the system. This becomes important in order to distinguish direct input from indirect input (see Section 2.2.2 below), as well as additional factors described in the taxonomy of location presented in Chapter Three.

2.2.1 Location and performance

In order to evaluate and compare user interface designs, HCI researchers have long attempted to create models of user performance.

Fitts's Law

One model of user performance that has been particularly useful to HCI is Fitts's Law [20,47]. Fitts's Law estimates the movement time of a selection (MT) with respect to the distance to the target (A) and the width of the target (W):

$$MT = a + b \times log_2\left(2 \times \frac{A}{W}\right) \quad (2.1)$$

with *a* and *b* being empirically-determined constants [47].

When applied to tool containers, Fitts's Law provides two important observations. First, the smaller a tool's visual representation is inside a container (i.e., the tool's width, W), the longer it will take to select. Second, the farther a tool container is from the user's current point of interaction (this distance is called the amplitude of movement, A), the longer it will take to select a tool from it. One important observation is that this distance, A, is entirely defined by the location of the tool container and the location of a user's point of interaction.

Fitts's Law was originally developed for 1D reciprocal tapping tasks [20]. Fitts and Peterson further extended this work to discrete tapping tasks [21]. Subsequent research has extended Fitts's Law to 2D selections in computer environments [47], which has proved quite useful for HCI research. It has also been shown to hold for many different input devices, including the mouse, touch, and eye gaze [46], and has been used to evaluate and compare many interaction techniques.

In-place tool containers

Fitts's Law provides an important characteristic for tool containers. Reducing the distance between the current point of interaction and the tool container will reduce the time required to select from it, so tool containers that are displayed above the user's point of interaction (or "in-place containers") will be faster to select from then tool containers displayed anywhere else in the interface. This observation is not novel, and has been the motivating factor for many techniques from the literature [10,15,28,29,43,50,71].

Representation location and awareness

As noted in Section 2.1, tool containers have a visual representation, and the location of this representation can affect aspects of user performance. For example, Price et al. [55] investigated three representation locations in a tangible learning environment. In their study, they built a digital tabletop system to teach children about the behavior of light. The tabletop was augmented with physically tangible objects, such as lights and mirrors, which affect the virtual environment.

They define three feedback representation locations: discrete, co-located, and embedded, defined as completely separate from the tangible (on a wall display), adjacent to the tangible (on the tabletop), and within the tangible (displayed on a screen attached to the tangible), respectively. The authors found that awareness was negatively affected by the discrete location, because awareness "...becomes a hard task in this context, and impedes interpretation of 'interference' events" [55].

2.2.2 Reach

Reach refers to "…the maximum extension of the arm at which the user can perform a manipulation" [67]. In this context, reach is a physical manifestation of the direct manipulation interface. When provided with indirect input, this limitation is often no longer valid, because indirect input is not limited to the physical length of a user's arm.

Reach is important for a study of location, specifically in multi-user direct input applications, because reach defines the area where tool containers could be displayed for each user.

Direct input

Buxton defines direct input devices as "devices where input takes place directly on the display surface" [11]. In other words, direct input, by definition, implies that the input space and the display space are overlaid.

Examples of direct input include touchscreens and stylus input in Tablet PCs. The point of interaction for direct input is the physical interaction point. That is, the interface location where the physical contact occurs is also the location of the virtual point of interaction.

Indirect input

The converse to direct input is indirect input. Indirect input implies that the input space and the display space are distinct [11]. Indirect input typically offers a virtual embodiment as feedback of where the point of interaction currently is in the workspace. One of the most prominent input devices, the mouse, is a prime example of indirect input; the physical action of moving the mouse moves the point of interaction (the cursor) on the screen.

Reach in user interfaces

One limitation of direct input is reach [67]. Interface elements that are displayed out of a user's physical reach cannot be interacted with without physically moving around the space. Some interface elements may not be reachable at all; for example, a wall mounted display may be too tall for children to reach elements at the top of the display.

Reach is not often an issue for indirect input. Whereas direct input has a physical limitation (e.g., arm length), indirect input can access any element in the interface, even elements on another display located in a different room!

Within single-user applications, reach is rarely an issue. Single-user applications are developed specifically for one, often stationary, person and so all elements can be displayed within easy reach of this person. In multi-display environments, some elements may be physically out of reach of the user, but this is often mitigated with the use of an indirect input device.

On the other hand, when multiple users are working within the same interface, reach can be used as a signal to the ownership of that item [67]. For example, if one user places an item within their reach but outside of the reach of the other users, this may be a sign of the user taking ownership of this item. Some systems, such as TractorBeam [51] and HybridPointing [25], attempt to mitigate the problems with direct input on large displays by providing direct and indirect input in the same input device.

2.2.3 Splitting the interface into areas

Researchers have investigated two different methods of dividing a shared interface into parts. The most common method is territoriality, which defines areas on a tabletop display to be used for individual work and other areas for group work. Another method follows the workbench approach, where the user's input action is location dependent: a stroke in the cutting area will cut the artifact, whereas the same stroke in the drawing area will draw a line.

Territoriality

Tables are commonly used to organize individual work and to support collaborative work tasks [60]. Due to the important nature of tables in the real world, interface designers have been interested in augmenting real-world tables with digital displays and input devices. Scott et al. [60] state that digital tabletop applications should model real world tables because digital tabletops can support "...the interaction skills people have developed over years of collaborating at traditional tables".

There are many open questions in digital tabletop designs, but one aspect of the real world does appear to transfer to digital tables: territoriality. Territoriality has been described as the partitioning of a physical workspace into territories [60]. Each territory supports certain kinds of collaborative work. Scott et al. [60] have shown three types of territories: personal, group, and storage. Territoriality has also been investigated with respect to reach. In [67], the authors found the Reach Envelope model correctly predicted that the tabletop surface would be split into regions defined by the reach of the users: areas within the reach of a user defined their personal workspace, whereas areas beyond the limits of each user's reach defined the group workspace.

Another project that partitions the workspace is Caretta [65], but in this case they use multiple input devices and displays. In this system, a user's personal workspace is displayed on a PDA, allowing for the personal workspace to also be private. There is also a public space, a tabletop display, where all users can interact together.

The workbench approach

Instead of defining areas on the tabletop with respect to the users' locations around the tabletop, Modal Spaces [18] uses the workbench approach. Areas in the shared interface define the tasks and operations that can occur in that area. Users can move a document into one of the Modal Spaces to perform the actions that are valid in that location. This view of splitting the interface is document-centered, instead of user-centered.

2.3 ATTENTION

Attention is a topic that all humans innately understand: "Attention is the cognitive process of selectively concentrating on one aspect of the environment while ignoring other things" [2]. In cognitive psychology, attention is currently understood as a broad array of phenomena. Researchers often divide attention into five types: focused, selective, switched, divided, and sustained [70]. In this thesis, I am specifically interested in focused and selective attention.

There are at least two input channels to attention models: visual and auditory. Much of the initial research into attention focused on auditory attention [14]. In this thesis, I am interested in visual attention. In this section, I first describe the different types of attention, focusing on the two that are relevant to the work in this thesis: focused and selective attention. Next, I move to a more specific kind of attention, visual attention, followed by a discussion of visual search. I conclude this section with a discussion of attention in the HCI literature.

2.3.1 Types of attention

In this thesis, I am interested in how attention is affected by tool containers displayed at different locations. Tool containers can be displayed over the user's point of interaction. This point is often referred to as the focus point [70], and attention at this location is called focused attention. When users make a selection from a tool container not displayed at their current point of focus, users must shift their attention to the tool container's location. This shift requires users to focus on a different part of the environment, and this type of attention is called selective attention.

Focused attention

Focused attention requires a conscious and sustained effort to attend to only a portion of the environment [70]. In visual attention, the center of the area attended to is called the focus (see Figure 12). There is a cost to maintaining attention to the same location, such as the sustained attention required by a night-watchman [70]; however, it is in shifting one's attention that many problems occur, such as being interrupted and forgetting previous information that was valid [3]. Voluntarily moving one's attention from one focus point to another is called selective attention.

Selective attention

Selective attention requires attention shifts from one channel or task to another [70]. An attention shift incurs an attentional cost on users, and as attention is typically thought of as a limited resource [70], more expensive attentional shifts will affect a user's performance more. Even as one attempts to focus on one area of the environment, attention may be inadvertently shifted to other areas. For example, hearing footsteps coming towards you will draw your attention from

the book you are reading. In this example, the two stimuli are in different input channels: the footsteps are auditory and reading the book is visual. Even though the footsteps are in a different channel, you may still shift your visual attention from the book to the direction of the footsteps.

The power of selective attention can also be demonstrated by the example above. If the footsteps are regular, you may assume it is a runner and not bother to look over; however, you may look over at the footsteps if they are accompanied by calls for help.

2.3.2 Visual attention

There are two related models for visual attention: the spotlight model, and the model that supersedes it, the zoom-lens model [70].

Visual attention is split into two stages: in the first stage, the entire scene is processed in parallel, and in the second stage, attention is focused to areas of the scene in a serial fashion [70]. The area of the environment in which the user focuses (see Figure 12, the focus) is where most information is gathered from the environment. Artifacts that appear in the fringe are pre-attentively processed (that is, processed before users consciously expend effort [70]), but the user may not be consciously aware that they have gathered any information from this area.

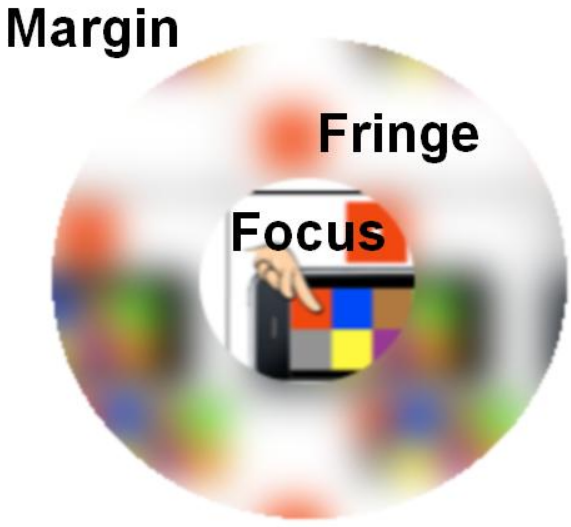


Figure 12 - Visual field of selective visual attention

2.3.3 Visual search

Visual search is “an effort to detect or locate a target item whose presence or position within the search field is not known a priori” [70]. Users can focus on one element out of a set of distracters using selective attention, and focusing on this element requires first acquiring this element through visual search. Visual search is one of the core activities in typically GUI applications [24], as users are required to search for the graphical element they want to interact with.

Researchers have used the response time (RT) or accuracy as measures of how difficult a visual search task is [70]. When the element users are looking for is dissimilar to the other elements in the search set, the RT is essentially constant: elements that are different from the distracters around it “pop out” to users’ eyes, making the search task trivial. On the other hand, when all of the elements are highly similar, RT increases almost linearly with set size [24]. This suggests that visual search is a serial process, since users must look at each item individually and decide if this is the item they are searching for or not.

Treisman and Gelade suggested that visual attention has both a pre-attentive aspect and a conscious attentive aspect. The feature integration theory [68] states that complex objects are first interpreted pre-attentively by their simple attributes, such as colour, shape, and motion, creating a set of feature maps representing the items in the environment. Next, focusing on one particular area of the environment allows a user to consciously interpret what they are looking at.

Visual attention and visual search have been investigated in HCI, typically in menu search [19] and icon search [24].

2.3.4 Attention in HCI

Attention and cognitive load are often motivating factors in much HCI research [e.g., 10,23,28,29,40,41,42,48]. Attention is often not defined, simply relying on the colloquial definition of attention that all people intuitively understand.

Some models of attention and human performance do exist, including ACT-R [7], Eye Movements and Movements of Attention (EMMA) [57], and Executive Process-Interactive

Control (EPIC) [6]. These models of attention have been used to study visual search in user interfaces [24]; however, in this thesis, I am not interested strictly in visual search, but in the attentional costs of switching attention, which requires visual search to acquire the target that visual attention is switching to.

Shifting attention

One model of shifting attention has been proposed by Plumlee and Ware [53]. In this research, the authors investigated the time required to compare two items that were not necessarily visible at the same time, thus requiring the users to scroll or zoom to view the two items. They proposed that the time to compare the items is:

$$T = S + \sum_1^v (B_i + D_i) \quad (2.2)$$

where S is the setup time before the comparison, v is the number of switches between the two items, B is the time spent switching between the two items, and D is the time spent at that item.

According to this model, we can reduce the comparison time by reducing the number of switches between the two items, reducing the time to travel between the two, or reducing the time spent at each item. Applying this model to tool container selections, I am not interested in D (that is, how users collect information from the tool container). In this thesis, I look at reducing B - the time spent between the items.

2.4 AWARENESS

In this section, I turn my attention to one important aspect of collaborative work: awareness. I first define awareness and different types of awareness that have been introduced in the CSCW literature. Next, I move into a discussion of how to measure awareness, and conclude with some factors that affect awareness in collaborative tasks.

2.4.1 Definition and types

Similar to the idea of attention, awareness is something that everyone understands. We are aware of the other cars on the road, of our kids playing in the backyard, and of those footsteps approaching from behind us; however, in the CSCW literature, there is debate around the definition of awareness. For example, Schmidt makes the case that awareness is not a single idea: there are lots of types of awareness, or as he writes: "...of what are actors supposedly aware when we in CSCW use the term 'awareness': awareness of what?" [58].

Awareness has been generally accepted to be the "understanding of the activities of others, which provides a context for your own activity" [16]. It was originally studied in ethnographic studies, including subway control rooms [35] and air traffic control rooms [45], and has more recently moved into the experimental laboratory [33,37]. In the CSCW literature, different types of awareness have been defined and investigated: situation awareness and workspace awareness are the two main types of awareness.

Situation awareness

Situation awareness is the "up-to-the-minute cognizance required to operate or maintain a system" [3]. Much of the research into situation awareness is based on observational studies in aviation.

Workspace awareness

In this thesis, workspace awareness is the most relevant type of awareness. Workspace awareness has been defined as "the up-to-the-moment understanding of another person's interaction with a shared workspace" [32]. It includes "…events happening in the workspace – inside the temporal and physical bounds of the task that the group is carrying out" [32], and so workspace awareness includes both awareness of the state of the workspace itself, as well as the interactions within it [32,37].

Ha et al. [33] describe a type of awareness called awareness of action. Awareness of action is a part of workspace awareness, since the interactions with the shared workspace can be considered

with respect to each action each user takes. In this thesis, I am interested in a type of awareness of action: selection awareness. Whereas awareness of action could include interactions such as drawing, selection awareness includes only the GUI tool selections that occur in the workspace.

2.4.2 Measuring awareness

Awareness is difficult to measure directly. One approach is instead to measure the lack of awareness by observing conflicts with the aspects of work that should be supported by good awareness [37].

Much previous work has attempted to measure awareness by completion time and errors [33]. Following Gutwin and Greenberg's [30,32] characterization of workspace awareness, Hornecker et al. [37] describe that "clashes, collisions, and breakdowns" are examples of a lack of awareness. They call these negative awareness indicators interference, or "unintended negative influence on another user's actions" [37].

2.4.3 What affects awareness?

With a general understanding of what awareness is and how to measure it, researchers have investigated some aspects of workspaces that affect awareness. In this section, I describe two aspects that have been shown to affect awareness: presence and input type.

Presence (collocated or remote)

Collocated teams use social cues and social dynamics to coordinate collaborative work [35,45]. The affordances that are so natural in the collocated setting are not available in distributed (or remote) workspaces. For this reason, much work in CSCW has been in supporting distributed work [e.g., 32,66]. In this thesis, I am interested solely in collocated applications, and focus on understanding awareness as a baseline before attempting to augment distributed systems.

Input type

Ha et al. [33] compared mice, stylus, and touch input in a collocated tabletop application. They found that input type (direct input or indirect input) affected awareness of action: direct input

(touch) provided faster response from other users, showing that direct input was better than indirect input for awareness of action [33].

Other researchers have shown that input type affects awareness. For example, Hornecker et al. [37] found that touch input "…resulted in more positive indicators of workspace awareness", but also caused more interference than mouse-based input. They also note that although touch input caused more interference, the recovery from these collisions was faster than with mouse input, suggesting that the physical actions afford more fluid recovery from collisions.

2.4.4 Tradeoff between supporting group work and individual work

In multi-user applications, users often switch from working individually and working as a group [31]. Humans can only effectively attend to one thing at a time (see attention in Section 2.3), and because applications must support both the individual work and the group work, there is then a tradeoff between supporting the individual work and the group work.

One important aspect to supporting effective group work is supporting proper group awareness (see awareness above); however, some techniques which are powerful for individual users, such as keyboard shortcuts, do not provide good awareness of the action to other users [31,65] because the physical movement is small, and there may be no obvious change in the workspace.

There is also an important issue of supporting the transition between working as an individual and working as a group. This was one of the design guidelines of digital tabletop collaborative design proposed by Scott et al. [59]. In this thesis, I do not focus on effectively supporting these transitions; instead, I provide a baseline for performance in an awareness task, which provides context for future work in augmenting collaborative tabletop systems to better support group work.

CHAPTER 3

A TAXONOMY OF TOOL CONTAINER LOCATION

Tool containers are a common way of organizing an interface by grouping tools into one visual representation. Most WIMP-style GUI interfaces make use of tool containers, with common examples including toolbars, contextual windows, and menus. More complex tool containers can be found in the HCI literature, such as marking menus [42] and Toolglasses [10]. These containers have some properties which differ from simple tool containers like toolbars and menus; for example, they may use gestures or bimanual input.

It is difficult to compare different tool containers because of their various properties; however, all tool containers share one property. Tool containers must have a visual representation, and thus, tool container location is a common property among all tool containers.

To organize my investigation of tool container location, I propose a taxonomy that compares three conceptual locations (in-place, inside workspace, outside workspace) against various performance issues and properties that affect the performance of tool containers at each of the locations.

In this chapter, I first describe the three conceptual locations and how I define a workspace. I then describe each of the performance issues and properties, and how each affects tool containers displayed in the three locations.

3.1 THE THREE CONCEPTUAL LOCATIONS

There are three conceptual locations where tool containers can be displayed: in-place, inside workspace, and outside workspace. In order to describe the three conceptual locations, I must first describe how I define a workspace.

3.1.1 Workspace

Workspaces contain the tools we can use, as well as the work artifacts (such as documents, images). A workspace can be a dynamic concept, but we often think of a workspace as the area in which we are interacting. In computing environments, the workspace is limited to the available screen space. This means that the workspace cannot be extended outside of the main display to the walls or the floor unless there are additional displays there; however, which parts of the available screen real estate are part of the current workspace?

Most current computer applications are designed for single-user, single-display, mouse and keyboard setups. In these applications, the application window in which the user is currently working can be thought of as the workspace. A typical task in this type of environment is word processing. The work artifact the user is working on, in this case the document, is in the center of the workspace with the tools stored along the edges of the screen (see Figure 13).

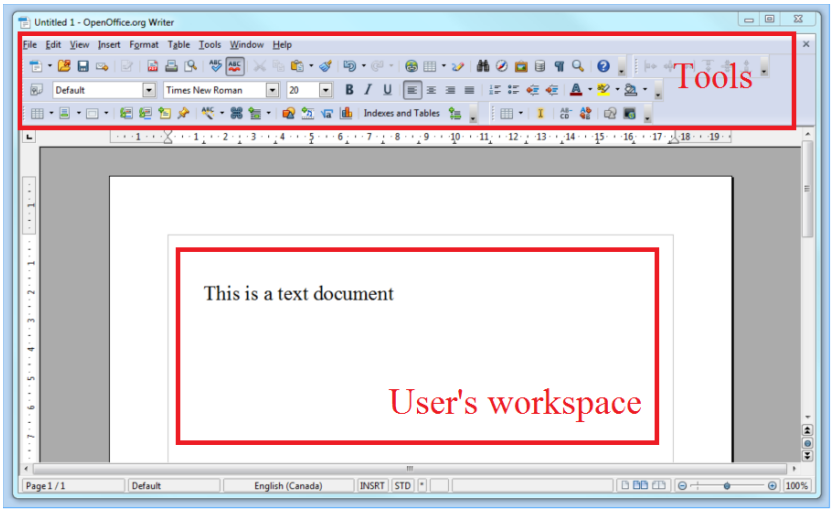


Figure 13 - OpenOffice window showing tools along the top, workspace in the center

Another type of computing environment is one where multiple users are working on a shared display. These displays are often horizontal, forming digital tabletop environments with users distributed around the edges of the table. In this situation, the workspace of an individual user is often not the entire display. Territoriality describes how users partition a display into personal workspaces and group workspaces (see Figure 14).

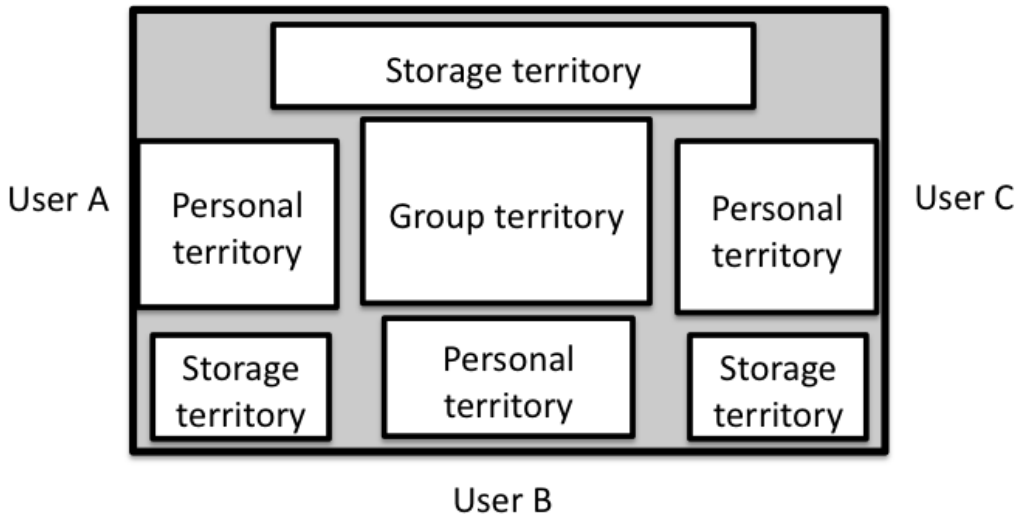


Figure 14 - Territoriality divides the table into territories

For the single user, dual-display setups are becoming commonplace. The additional display can be included in the workspace by extending the window over both screens. On the other hand, this additional display could be a separate workspace, say containing an e-mail client. For multi-user tabletop applications, the user's workspace is typically the area directly in front of them; however, two users could merge their workspaces together, creating a larger workspace for the two to perform some collaborative task.

3.1.2 Locations for tool containers

With the above discussion of workspace, there are three conceptual locations in which we could display a tool container: in-place, inside workspace, and outside workspace.

In-place

An in-place tool container is displayed near the user's point of interaction inside their workspace. According to Fitts's Law, these containers should be the fastest to select from because they are closest to a user's current point of interaction. In-place containers are often popups, such as right-click context menus, but they may have static visual representations as well, such as Toolglasses [10].

Inside workspace

The second conceptual location is inside the workspace. The most common examples of these types of containers are toolbars, hierarchical menus, and floating palettes.

Outside workspace

The third conceptual location is outside the user's workspace. This location is not very commonly used as a location for tool containers, and is typically only used as notification of system events, such as popups in Windows from the system tray.

3.2 A TAXONOMY OF TOOL CONTAINER LOCATION

In Table 1, I present a taxonomy of tool container location. This taxonomy divides tool containers by the conceptual locations in which they can be displayed: in-place, inside workspace, and outside workspace. The taxonomy also describes seven performance issues and tool container properties that affect user performance for tool containers displayed in the conceptual locations. Each of the seven performance issues and properties are discussed in this section.

The taxonomy provides context for the two empirical studies in Chapter Four and Chapter Five, as well as future work, described in Chapter Seven.

The table rows have been split into two parts: performance issues and optional tool container properties. The first part lists four tool container performance issues that occur with all tool containers: visual shift, visual search, occlusion of the workspace, and association of tool and workspace. In the second part, I list three properties that some tool containers have: popup, bimanual interaction, and the location of interaction. All tool containers are affected by the items in the first part, whereas the items in the second part only affect some tool containers (for example, only some tool containers are popups, but we can compare all popup techniques in the three conceptual locations).

Table 1 - A taxonomy of tool container location

	In-place	**Inside workspace**	**Outside workspace**
Performance issues			
Visual shift	-Smallest visual shift	-Medium visual shift	-Largest visual shift
Visual search	-Easy to acquire the container	-Clutter may make this difficult; e.g., to select a tool in stacked toolbars first requires users to acquire the toolbar before the tool	-May require acquiring another device
Occlusion of the workspace by container	-In-place containers obscure workspace near the point of interaction	-Floating containers obscure workspace, away from point of interaction -Static tool containers reduce usable screen real-estate	-By definition, do not obscure the workspace
What workspace object does this tool affect?	-In-place containers are often contextual, and modify the selected item in the workspace -Owned by user who owns the workspace	-Associated with this window/application -Owned by user who owns the workspace (multi-user setup)	-May not affect the workspace at all (e.g., the Start menu) -Owned by many users, or no one (group tools)
Optional tool containers properties			
Popup	-Fast selection -Most popup techniques use this location	-Modal dialogs	-These are often notifications – e.g., IM notifications
Bimanual interaction	-Commonly, use non-dominant hand (NDH) to position the tool; occlusion by NDH	-A tool can be selected with NDH, possibly parallelizing a task, but could cause occlusion	-NDH does not occlude workspace -Keyboard shortcuts are the most common
Location of interaction (direct and indirect input)	Direct - Occlusion from the interaction device (stylus or finger) Indirect - Does not require clutching	Direct - Less likely that interaction will cause occlusion Indirect – May require clutching to acquire	Direct -No occlusion from interaction device Indirect - Displayed along the edges of screen provides infinite width containers

The three conceptual locations allow us to distinguish between different performance issues that may occur with tool containers displayed at these locations. The following discussion takes each row from Table 1 and describes how this performance issue affects user performance of selecting a tool from tool containers displayed in that location. I also provide examples from the literature of techniques that exhibit this user performance issue.

3.2.1 Visual shift

Users have a point of interaction in the workspace. This point of interaction is also called their point of focus, because this is often where users are looking. To select a tool from a tool container, users must first visually acquire the tool container. The visual shift required depends on where the tool container is displayed: the farther the tool container is from the current point of focus, the longer it will take to shift visual attention to the tool container.

Tool containers displayed in-place require a very small visual shift, whereas containers displayed inside the workspace require a medium visual shift (assuming the workspace is relatively small), and containers displayed outside the workspace usually require the largest visual shift.

3.2.2 Visual search

Related to visual shift is the issue of visual search. The visual shift occurs first, followed by a visual search of the newly acquired workspace area. In this step, users search the newly acquired area for the correct tool container that contains the tool they wish to select.

Tool containers displayed over the current point of interaction require little visual searching because the tool container likely stands out from the work artifacts (i.e., it is easy to see a toolbar over a text document because it looks very different) - see Figure 15.

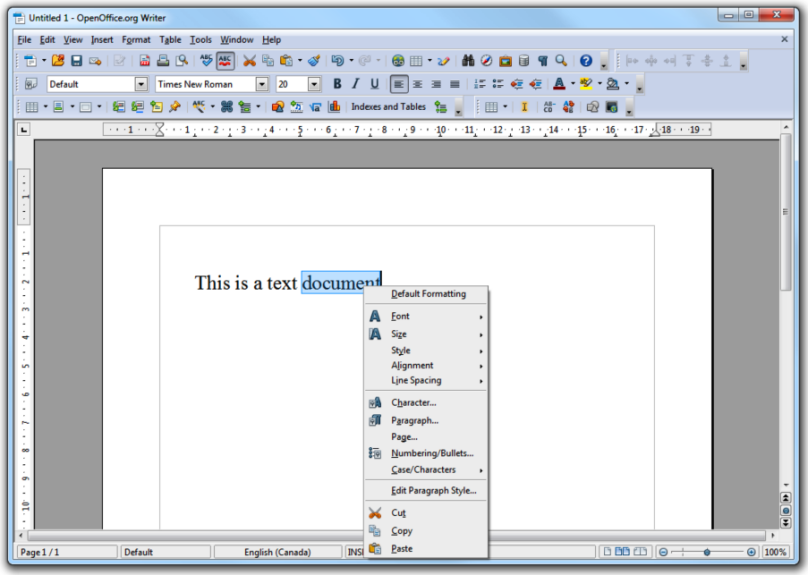


Figure 15 - OpenOffice with in-place popup tool container

Tool containers in the second location, inside workspace, require additional searching, especially when the application has multiple tool containers displayed around the workspace. For example, Figure 16 shows an application with multiple toolbars anchored to the top of the application window. To select the highlighted tool, users must first acquire the correct toolbar, and then select the correct item within it.

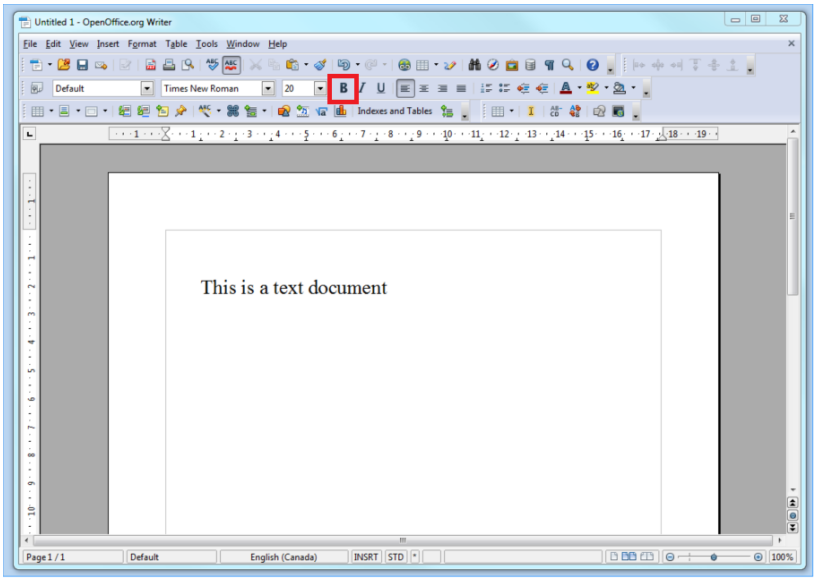


Figure 16 - OpenOffice with one tool to select from stacked toolbars along the top

Tool containers displayed in the third location, outside workspace, may not even be displayed on the same display as the workspace. For example, in LensMouse [71] and Visual Trackpad [15] (Figure 17), a toolbar is displayed on an external direct touch input device. These tool containers require users to acquire a separate input device before they can select the correct tool. Whether they are displayed on the same display or not, they still require a larger visual search than simply searching within the current workspace.

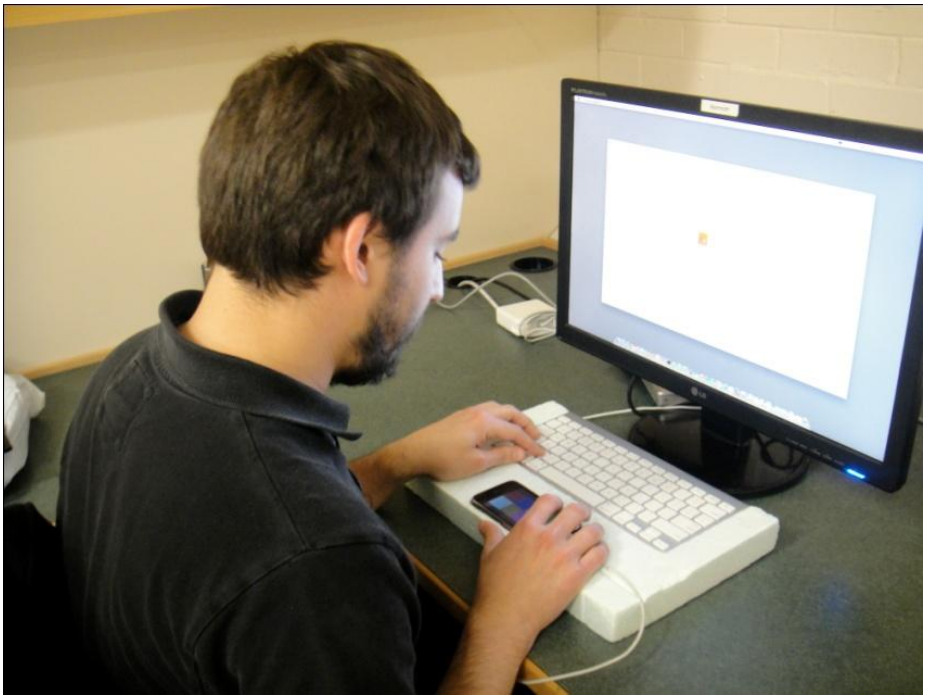


Figure 17 – Selecting a colour from palette in Visual Trackpad prototype

Note that the visual search discussed in this section is not the search required to find the correct item inside the tool container. Regardless of the location of the tool container, the visual search for the correct item inside the container is the same. The issue of creating tool containers such that items are easy to locate inside them is outside the scope of this thesis.

3.2.3 Occlusion of the workspace by the container

Since a tool container must have a visual representation, there is a chance that it may occlude some of the user's workspace; however, some areas of the workspace are likely more important than others.

In-place tool containers, by definition, occlude part of the workspace that the user is currently interacting with. The workspace area around the point of interaction may be integral to the task the user is trying to accomplish, and so obscuring it may be costly for users, causing them to forget their current task.

Tool containers displayed in the second location, inside workspace, can be of two types: static location or moveable (see Figure 18).

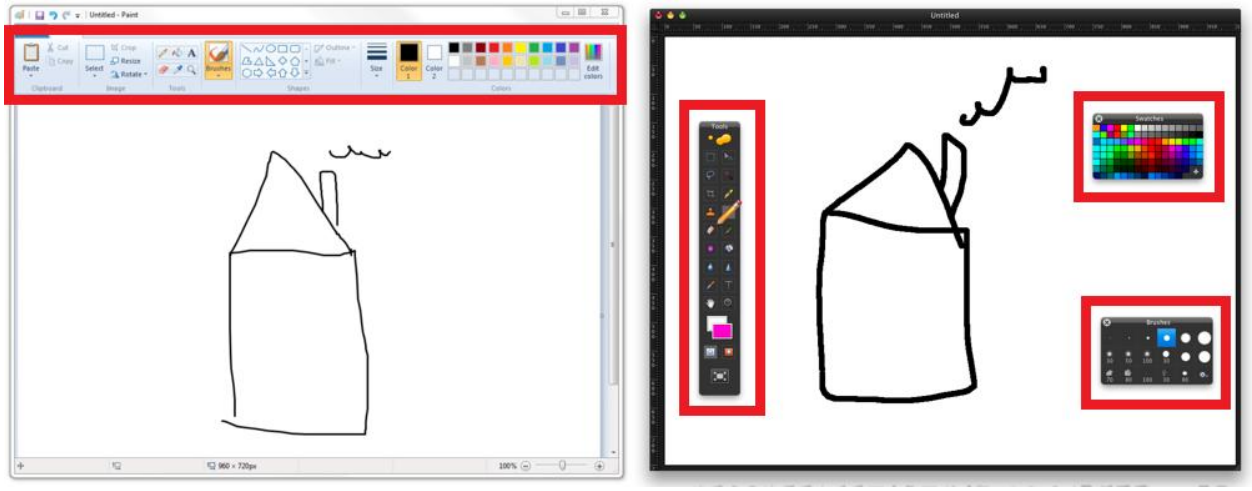


Figure 18 - a) static inside workspace toolbar, b) moveable inside workspace palettes

Examples of tool containers with a static location and that are displayed inside the user's workspace include toolbars and menus. These tool containers do not occlude the workspace, but do reduce the available screen real-estate for the workspace. This is one motivation for using popup tool containers instead of static tool containers: the popup tool containers free some screen real estate when they are not in use.

Examples of moveable tool containers that are displayed inside the user's workspace include floating toolbars and palettes. These tool containers do occlude some parts of the workspace, but because the location of these tool containers is user defined, users can organize them in such a way as to maximize their utility for their current task and minimize the tool container's effect of occluding important parts of the workspace.

Outside workspace tool containers, by definition, do not occlude any of the workspace.

3.2.4 Association of tool inside the container

A tool stored in a tool container can have three effects. First, a tool can have an immediate workspace effect. For example, a user can highlight some text and then click the bolding tool, or select Quit from an application's menu. Second, a tool can affect the workspace action that will follow it. For example, a user can select the circle tool and then draw a circle. Third, a tool can cause some global system effect, such as when users launch an application from the Start menu.

The location of a tool container can indicate to the user how the tools contained within it are associated with the system. In this thesis, I am interested in two aspects of association: workspace item association, and workspace and user ownership.

Workspace item association

Contextual tool containers can be located in any of the three locations, but the most common contextual tool container is the in-place popup tool container. This is likely because in-place contextual containers are more easily associated with the workspace item that they affect. There are, however, examples of contextual tool containers displayed inside the workspace. A recent example is the Ribbon in Microsoft Office 2007 (see Figure 19). In most usage, the Ribbon is a static tabbed tool container, but when certain items are selected in the workspace (for example, a table), contextual tools are displayed in the Ribbon (such as adding and deleting rows).

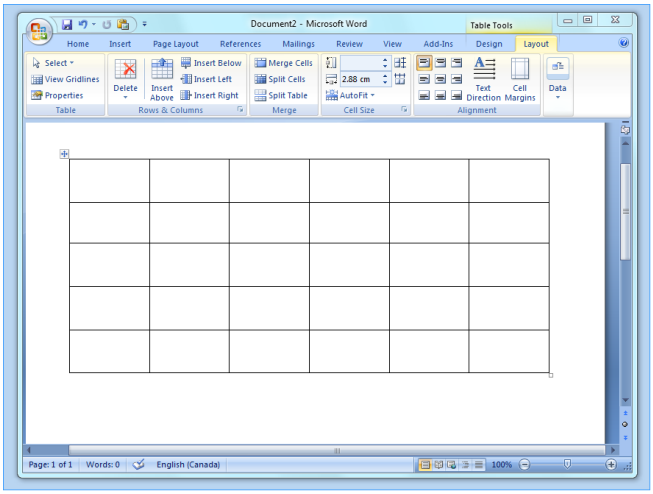


Figure 19 - Contextual table tools in the Ribbon (MSWord 2007)

Examples of contextual tool containers displayed outside of the workspace are LensMouse [71] and Visual Trackpad [15]. For these tool containers, it may be difficult for users to know which workspace item is going to be affected by tool selections in the containers, because there is no visual link between the tools in the tool container and the workspace item it will be affecting.

Workspace and user ownership

In addition to knowing which workspace item will be affected by a tool, there are two other association issues. Users must know which workspace this tool container is associated with, and which users can access the tools inside the tool container.

The first issue is that of workspace association. In most modern operating systems, users are able to run multiple applications at the same time. In addition, these applications can have multiple windows open. For these situations, the question of which window/application a tool container affects becomes important.

Tool containers displayed in-place and inside the workspace are likely obvious: tools within the containers should affect the application and workspace that contain them. Tool containers displayed outside the workspace can cause confusion for users, since there is no link between the action of selecting the tool and the action that could occur inside of a user's workspace. However, since many tool containers displayed outside the user's workspace are notifications or global actions, they are not associated with any window/application. Designers should be careful to distinguish between tools that can cause global effects, such as launching an application from the Start menu, and workspace tools that are displayed outside of the workspace.

The second issue is that of user ownership. In multi-user applications, the location of a tool container can indicate which user has ownership of this tool container. For example, tool containers displayed in-place and inside a user's workspace should be owned by that user; other users should not attempt to access tools from this tool container. On the other hand, tool containers displayed outside of the users' workspaces are often considered group tools. Group tools consist of those tools that have global effects, such as closing the application, and tools that every user can access but aren't owned by any one user (i.e., shared tools).

3.2.5 Popup tool containers

A popup tool container has two modes. In the first mode, it is hidden from view. Once a modifier is activated, the container is in the second mode when it is displayed on the screen.

Popup tool containers are quite common [15,22,29,36,42,43,56]. One motivation for using a popup tool container is that the container does not require any screen real estate until users need it. This implies, though, that users must remember how to activate the popup. As the number of popup containers increases, it may be more difficult for users to remember how to activate each of the containers, and which container contains the tool they are searching for. For this reason, most GUI applications offer only one type of popup container, often a right-click context menu.

The most common location for popup tool containers is in-place. These popup tool containers are fast to select from, because users don't have to move their point of interaction (as well as their point of focus) far. They are often contextual in the sense that the popup contains tools that will affect the object selected in the workspace (for example, right-click context menus). Tool containers displayed outside a user's workspace are typically used as notifications of events. For example, IM clients may pop up a notification of new messages at the edge of the screen (Figure 20). These are different than the tool containers discussed so far, but the notifications share properties of regular tool containers: they have a visual representation, and contain items that are activated by simple selections. Many of these IM notifications open a chat window when they are clicked, and also have a dismiss item (typically a close button).

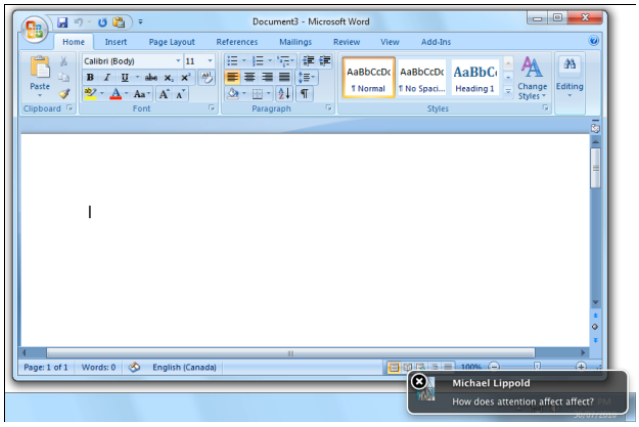


Figure 20 - IM notification outside workspace

Popup tool containers displayed inside a user's workspace are less common than those described above. These could be used as notifications as well; for example, in an IRC client, notifications could appear in the client list when a friend sends a message or when a new client joins the channel. Another example of a popup in this location is a modal dialog, such as a "Save as" or "Print" dialog.

3.2.6 Bimanual interaction

Some tool containers can be used bimanually, which is to say, in a way that make use of both hands. The most common way to offer bimanual input is to control the spatial position of the tool container with the user's non-dominant hand and make selections from the tool container with the user's dominant hand (e.g., Toolglasses [10]); however, other tool containers that typically do not offer bimanual input could be used with both hands. For example, on a touchscreen, a toolbar item could be selected with a user's non-dominant hand and actions in the workspace occur with the user's dominant hand. This could allow users to parallelize a task: for example, selection of a tool and workspace activation occurring in quick succession.

In-place bimanual tool containers may cause occlusion of the workspace by the non-dominant hand (or by another device, as with Toolglasses [10]). Inside-workspace bimanual tool containers reduce the occlusion from the non-dominant hand, as these tool containers can be displayed near the edges of the workspace where the non-dominant hand rests.

Outside workspace bimanual tool containers are not very common. One example of this type of tool container is an external touchscreen that houses a tool container. For example, there are external screens that also accept touch input (http://www.mimomonitors.com/). Tool containers could be displayed on this external touchscreen, and tool selections could occur from it with the non-dominant hand. One technique that could also belong to this category is keyboard shortcuts. Although a keyboard does not actually display a tool, the keys can represent GUI tools from the workspace. For example, clicking Ctrl+b in a word processor often accesses the bolding tool from the menu or toolbar. In this way, keyboard shortcuts can be considered to be bimanual, outside workspace, tool containers.

3.2.7 Location of interaction

The location of a tool container offers two types of locations. First, there is the location of the visual representation of the tool container. Second, there is the location of interaction when selecting tools from the tool container.

It is important to make the distinction between direct and indirect selection mechanisms. Direct input implies that the location of the interaction is the same as the location of the visual representation of the tool container. Indirect input is the opposite: the locations are not the same.

Direct input in a tool container may cause occlusion of the tool container and surrounding workspace by the interaction device (finger or stylus). On the other hand, indirect input typically offers a virtual embodiment, such as the ubiquitous mouse cursor. This embodiment causes little occlusion of the tool container and workspace.

Direct input is only affected by the three locations with respect to the occlusion of the workspace and tool container with the interaction device: in-place has highest occlusion, inside workspace has the potential for occlusion, and outside workspace has the least potential of occlusion.

Indirect input, however, does have additional issues in each of the locations. In-place indirect tool containers can often be selected without clutching (that is, moving the input device without moving the on-screen cursor), whereas inside workspace and outside workspace containers may require clutching to acquire. Another important observation is that inside workspace and outside workspace tool containers may be displayed along the edges of the screen, thereby creating infinite-width tool containers (see Fitts's Law in Section 2.2.1). Although in-place tool containers are fast because they are close, infinite width tool containers are fast because users do not have to worry about overshooting the tool, and so can access the tool container with one large ballistic movement. This is the reason why Mac OS X menus are considered faster than their Windows counterparts [1], a claim supported by Fitts's Law (covered in Section 2.2.1).

3.3 CONCLUSION

In this chapter, I proposed that the location of a tool container affects user performance of selections of the tools from within it. To begin investigating how location affects tool selections, I first described the concept of a workspace. I then outlined three conceptual locations (in-place, inside workspace, and outside workspace) where tool containers can be displayed. Finally, I described four performance issues and three tool container properties, and how theses aspects of tool containers are affected when they are displayed in the three conceptual locations.

The seven issues described in this chapter cover many of the design decisions behind tool containers; however, there are likely other performance issues that are affected by the three conceptual locations.

CHAPTER 4

IN-PLACE TOOLBARS AND INDIVIDUAL ATTENTION

In Chapter Three, I described a taxonomy of tool container location and how different performance issues and tool container properties are affected by tool containers displayed in the three conceptual locations. As described in Section 3.2.7, tool container location has two parts: the location of the visual representation and the location of interaction.

According to Fitts's Law, tool selections from in-place tool containers are faster than from tool containers displayed at either of the other two locations in the taxonomy. In this chapter, I am interested in providing an understanding of the attentional costs of in-place toolbars, but not necessarily those with in-place visual representations; instead, I am interested in in-place interaction tool containers.

An in-place interaction (IPI) tool container is a tool container that allows selections to occur without moving the interaction point from the workspace. That is, users can keep their place in the workspace while making a tool selection, and then can immediately continue their previously halted task. I call these types of selection mid-task selections.

One example of a mid-task selection is filling an area in a drawing program. A user notices that there is a small white patch in their drawing, so decides to fill it in with the colour surrounding it. This requires that the user select the area, then choose the fill colour from a palette. Static, and even floating, colour palettes require users to move their point of interaction (i.e., their mouse cursor) from the white patch, select a colour, and return to the area to be filled. Reacquiring this area is a costly operation, especially when the area to reacquire is small.

As can be seen from this example, the ideal solution would be a tool container that provides in-place interaction selections, so that after the selection, the current task can continue without having to reacquire the area of the workspace. Although there are many tool containers whose visual representations are in-place, there are not many IPI tool containers. Keyboard shortcuts are one way to provide this type of interaction, but there are reasons why keyboard shortcuts are

not ideal (for example, they require that users to remember key combinations, and can provide only a limited number of shortcuts).

One observation is particularly important here. Even though these selections must be in-place interactions, the tool containers do not necessarily have to be displayed in-place.

The taxonomy presented in Chapter Three outlines some of the performance issues that could affect performance of IPI tool selections. Both visual shift and visual search are relevant to this discussion, since the location of the tool container does not have to be in-place. These two performance issues of tool container location directly affect the attention required of the user, specifically visual attention. Since users can only focus at one point at a time, tool container locations other than in-place require users to shift their visual attention from the workspace and back again. The shifting also requires users to search the newly acquired area for the tool container they are searching for.

Providing effective IPI selections, then, requires answers to the following questions:

- What are the attentional costs, in terms of visual shifts and visual search, associated with each of the three conceptual locations in the taxonomy?
- What is the effect of access method on attention?
- What are the attentional costs of switching between indirect and direct input?

To answer these questions, I designed three in-place interaction tool containers, one in each of the three conceptual locations of the taxonomy. My designs show that it is possible to design IPI tool containers without necessarily displaying them in the in-place location.

I then ran an empirical comparison of the three techniques and compared them to static toolbars, the most common implementation of toolbars. I found that: users performed faster tool selections from the IPI tool containers than static toolbars for mid-task selections; the access method for the IPI toolbars is an important factor of user performance; and displaying a tool container on an external input device was not as expensive as expected.

In this chapter, I first describe the three in-place interaction techniques and discuss the design decisions behind their implementation. Second, I outline the experimental task, and the empirical study I ran to compare the three techniques to static toolbars. Finally, I discuss the contributions of this study to the overall taxonomy of tool container location.

4.1 RESEARCH QUESTIONS

There are three research questions I want to answer in this chapter.

What are the attentional costs associated with each of the conceptual locations in the taxonomy?

As described in Chapter Three, the taxonomy outlines some of the user performance issues related to displaying a tool container in three conceptual locations: in-place, inside workspace, and outside workspace. In this chapter, I am interested in the issues surrounding visual shifts and visual search (see Sections 3.2.1 and 3.2.2). Specifically, I want to know what are the attentional costs associated with each of the three conceptual tool container locations in terms of the costs of visual shifts and visual search.

What is the effect of access method on attention?

In this chapter, I am interested in IPI techniques. These techniques allow selections from tool containers without losing the point of interaction in the workspace. One way to provide in-place interaction selections is to use a modal tool container. A mode switch will toggle between moving the system cursor in the workspace and allowing a selection to occur from a tool container. I call this mode switch the *tool container's access method*. In this chapter, I investigate three access methods, and evaluate their effect on a user's attention.

What are the attentional costs of switching between indirect and direct input?

Tool containers displayed on a touchscreen can provide IPI selections. Direct input can be fast if the targets are large enough [61], and touchscreens are often considered easy to learn and easy to

use; however, WIMP-style interfaces are often built for a traditional mouse, which provides indirect input to the system. To leverage the benefits of both direct and indirect input, I developed a novel interaction technique that overlays a direct touch tool container on top of the indirect input device (in this case, the indirect input device is a trackpad). Overlaying both direct and indirect input on the same surface should leverage the benefits of both direct and indirect input; however, users must switch their attention from using one input modality to using the other input modality. It is still an open question if this input modality switch negatively affects a user's attention, and so I investigate this question in this chapter.

4.2 THE THREE IN-PLACE INTERACTION TECHNIQUES

In this section, I describe the three in-place interaction techniques, how they relate to techniques from the literature, and some of the design decisions that guided their development. It is worth noting up front that the three techniques use the same toolbar and differ in the location of the visual representation of the toolbar, the access method, and the input type. Also, the three techniques all require a mode switch, which I implemented as a simple keyboard shortcut.

In the introduction to this chapter, I described a colour-filling task, where a user noticed a small white patch in their drawing, so decides to fill it in with the colour surrounding it. I first show in Figure 21 how users would accomplish this task using a static toolbar, then describe each in-place technique and how users would accomplish this task using the in-place techniques.

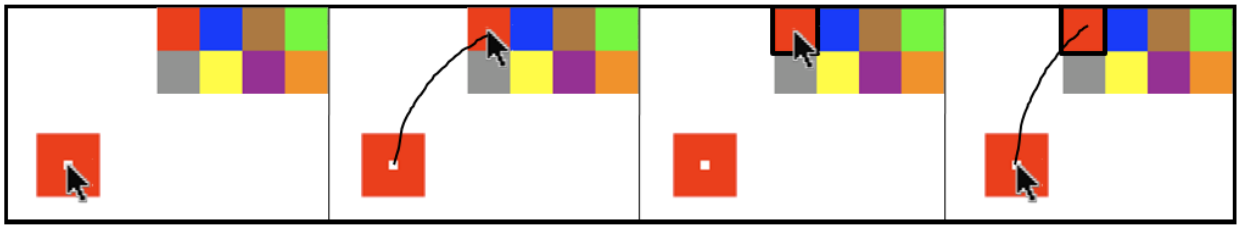


Figure 21 – Color-filling task using a static toolbar. In frame 1, the user has selected the region to be filled; in frame 2, the user moves to the static toolbar; in frame 3, the user selects the desired colour; in frame 4, the user moves back to the selected object and clicks.

4.2.1 Shadow Cursor (SC)

The first IPI technique is Shadow Cursor (SC). SC is an in-place popup toolbar (see Figure 22). Once SC has been activated by a mode switch key being depressed, the system cursor is hidden and a secondary cursor with a different visual representation (a hand pointer) appears in the toolbar. Users now have control of this secondary (i.e., shadow) cursor, and use it to make a selection from the toolbar. Once a toolbar item is selected, the toolbar is hidden, and the system cursor re-appears at the same location as when the mode switch key was depressed.

SC shares much of the same characteristics as popup tool containers (see Section 2.1.1). One tool container that it is particularly related to is the Hotbox technique [43]. Kurtenbach et al. [43] state that one motivation for developing the Hotbox was to clear the workspace of GUI widgets when they were not in use. This is the main motivation of most popup tool containers, coupled with the fact that popup tool containers are typically displayed close to the user's point of interaction, making selections from the tool container faster than tool containers displayed elsewhere in the interface.

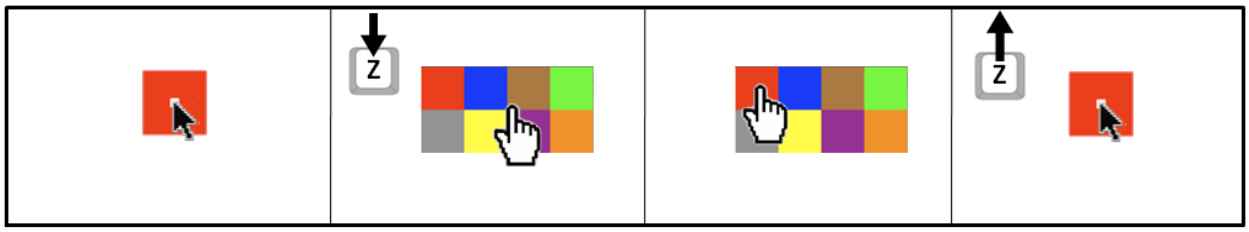


Figure 22 - Color-filling task using Shadow Cursor. In frame 1, the user has selected the region to be filled; in frame 2, the user presses the modifier key that pops up the toolbar; in frame 3, the user selects the desired colour using the shadow cursor; in frame 4, the user clicks in the target.

4.2.2 Warp Cursor (WC)

Static toolbars are one of the most common tool containers in current WIMP-style interfaces; however, one of their biggest limitations is that they are often anchored to the edges of an application's window, making them the tool containers the farthest away from the user's workspace. Traveling this pixel distance between the workspace and the tool container is,

according to Fitts's Law (see Section 2.2.1), a time consuming operation, compared to the time spent in the toolbar making a selection.

The second IPI technique is Warp Cursor (WC). WC is a modification of a static toolbar. Instead of requiring users to move their cursor over the pixel distance between the toolbar and the workspace, WC replaces this movement with faster eye movement [46]. Depressing the mode switch key causes the cursor to warp to the center of the toolbar, which is anchored at the edge of the screen, as is typical for static toolbars. Users now have control of the cursor within the toolbar. Once a selection is made, the cursor warps back to its original workspace location.

WC is similar to the MAGIC pointing technique [72], but replaces the gaze targeting with an explicit mode switch. In contrast to the MAGIC pointing technique, however, the cursor always warps to a fixed location on the screen. Different hot keys could map to different toolbars, allowing users to warp to the various tool containers on the screen while still maintaining the fixed warp locations. In Plumlee's and Ware's model [53], described in Section 2.3.4, this warping represents a reduction of B, which is to say, a reduction in the amount of time required to move between the targets.

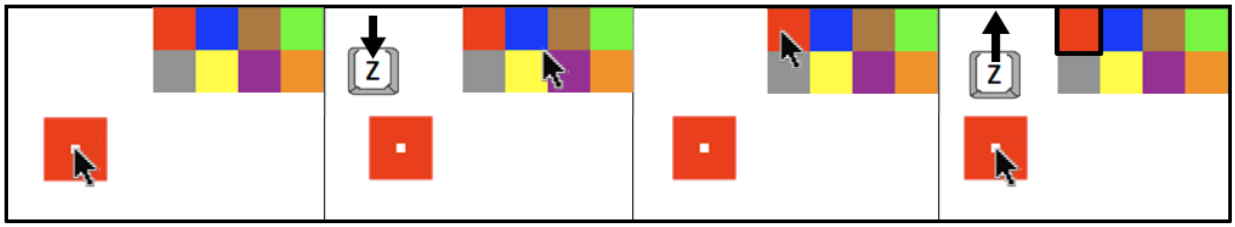


Figure 23 - Color-filling task using Warp Cursor. In frame 1, the user has selected the region to be filled; in frame 2, the user presses the modifier key that warps the cursor to the static toolbar; in frame 3, the user selects the desired colour; in frame 4, the user clicks in the target.

4.2.3 Visual Trackpad (VT)

The third IPI technique is a novel input device and technique: Visual Trackpad (VT). VT offers both direct and indirect input (see Figure 24), because it overlays a display surface over a regular indirect input trackpad, such as those often found on laptop computers. It is a trackpad in indirect

mode and a touchscreen in direct mode. Since VT overlays the two input modalities on the same input surface, it can combine the benefits of both.

In regular usage, VT functions the same as a regular laptop trackpad. Finger movement on the surface moves the system cursor and taps are interpreted as clicks. When the mode switch key is depressed, a toolbar is displayed on the trackpad's screen. Users can directly tap on a toolbar item, causing the toolbar to disappear, and VT returns to being an indirect input device. Although Visual Trackpad requires users to move their attention to an off-screen toolbar, touching a toolbar item directly is known to be fast [61], and may capitalize on muscle memory.

During the development of VT, I discovered that this technology is now available commercially (http://www.sharp.co.jp/mebius/products/pcnj70a/index.html). In addition, a similar technique was developed at the University of Manitoba. Instead of using a trackpad for indirect input, LensMouse [71] is a regular indirect input mouse with a touchscreen attached to the top. LensMouse and Visual Trackpad are closely related, since they both allow direct and indirect input in the same input device; however, they differ in one important aspect. VT requires the users to switch between direct and indirect input on the same input surface, whereas LensMouse requires the switch between direct and indirect input using two different parts of the input device. Stated another way, LensMouse allows indirect input by the physical movement of the input device and direct input on the top of the mouse, whereas VT allows indirect input in the same way that it allows direct input (that is, by the movement of fingers on the surface).

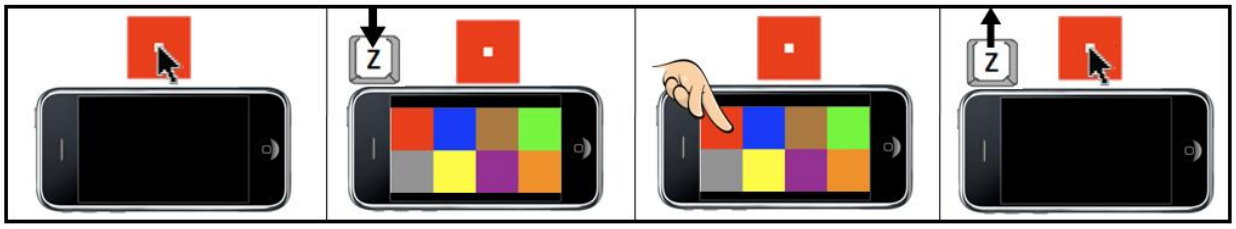


Figure 24 - Color-filling task using Visual Trackpad. In frame 1, the user has selected the region to be filled; in frame 2, the user presses the modifier key that pops up the toolbar on the trackpad's touchscreen; in frame 3, the user selects the desired colour using direct touch; in frame 4, the user clicks in the target.

4.2.4 Design considerations

There are three design considerations that guided the design of the three IPI techniques.

Mode switching

The first design consideration is that of mode switching. The three IPI techniques are modal, in that users switch from indirectly controlling the system cursor, to making selections from the tool container, and back to indirectly controlling the system cursor. Mode switches are often considered confusing for users, but constant physical action can reduce the cost of the mode switch [63].

The three containers in the study are activated via a constant physical action. They are activated with a press-and-hold of a keyboard shortcut, which means that the key must be held down during the entire selection from the tool container.

Conceptual locations and the techniques

The three techniques are displayed in the three taxonomy locations. SC is displayed in-place, WC is displayed inside the user's workspace, and VT is displayed outside the user's workspace. The location of interaction, on the other hand, is always in-place, because the three techniques allow users to make a selection from the tool container without moving their point of interaction from the workspace. These three techniques were used in an empirical study (see Section 4.3) to investigate how tool containers displayed in the three conceptual taxonomy locations affect individual attention.

Comparison of the three in-place techniques

The last design consideration is how the three techniques relate to each other. SC and VT are similar techniques, in that they are both popup tool containers; however, SC is displayed above the user's current point of interaction (the in-place display location), whereas VT is displayed on the trackpad (the outside workspace display location). This difference is important because VT requires a much larger visual shift, requiring additional attention to acquire the toolbar. VT also

requires users to shift their attention to a different input modality (direct input instead of indirect input). In a sense, VT is a direct input version of SC that requires a larger visual attention shift.

4.3 STUDY

In order to answer my three research questions (see Section 4.1), I ran an empirical comparison of the three in-place interaction techniques and compared them to static toolbars.

Note that in this study I investigate the differences between the different locations in the taxonomy presented in Chapter Three; however, each location has an associated access method (in-place popup, on-screen cursor warping, and external display direct touch). This means that location and access method are linked, and not fully crossed, in this study. I chose the combinations that seemed to make sense for interface design. Some of the crosses would likely never occur in real-world interfaces. For example, an in-place or inside workspace direct touch toolbar is unlikely to be a good design when coupled with a mouse; it is unlikely that users would want to move the cursor using a mouse, and make selections by tapping on the main display (why not just make the selection with the cursor?).

4.3.1 Task

One type of selection is particularly time consuming using static toolbars: mid-task selections. A mid-task selection is one in which the user must stop their current task, make a selection from the tool container, and return to the same workspace location to resume their task. These types of tasks have also been referred to as tool palette round trips [22]. Static toolbars are particularly costly for these types of selections because users shift their attention and their point of interaction away from the workspace, make a selection, and reacquire the same location in the workspace. Reacquiring the same location can be extra difficult, especially for cluttered workspaces or locations that are quite small to reacquire.

The experimental task requires mid-task selections to complete. Users are presented with a coloured square (50px X 50px), with a small white square (7px X 7px) in the center. Users are

required to fill in the white square with the colour that surrounds it. The colours must be selected from the toolbar (see Figures 21-24 for each technique).

Users first had to acquire the small white square with the mouse cursor, after which they would hear an audible signal. They then had to select the correct colour from the toolbar, and reacquire the white square with the mouse cursor in order to click inside it to fill it with the correct colour. One example of where this task might appear in real work was described in the introduction. Users are scanning through their image and notice that an area was left white, and so they want to fill in this white area with the colour that surrounds it. In this real world example, users must first find the white square to be filled in, select the correct colour, and then reacquire the white square with their mouse cursor in order to fill it in with that colour. The experimental task requires the same steps.

4.3.2 Methodology

The study required participants to complete the experimental task with the three IPI techniques (Shadow Cursor, Warp Cursor, and Visual Trackpad; see Section 4.2), as well as with a control technique, Static Toolbars. Note that only the Static Toolbar technique required users to target the small white square twice; the other techniques returned the cursor back to the target, allowing users to complete the task simply by clicking again after the colour selection. Reacquiring the target the second time is one reason why static toolbars do not support mid-task selections well.

Participants

Sixteen right-handed participants aged 20-34 (mean 24.3) were recruited from a participant mailing list for the study. This mailing list is made up of volunteers from the general university population, including students and staff. Six participants were male. All were experienced computer users (> 2 hr/day), and all but one were experienced with a trackpad input device.

Data collection

The study used a single-factor repeated measures design, with the factor *toolbar technique* having four levels: Static Toolbar (ST), Shadow Cursor (SC), Visual Trackpad (VT), and Warp

Cursor (WC). Each technique was demonstrated for the participant, and they then completed eight training trials with each technique. In the testing session, participants completed trials at 64 target locations, with colour selections chosen randomly. Target locations and colour selections were standardized across the four conditions. The order of techniques was balanced using a Latin square, and the study took 45 minutes to complete.

The software collected four time measures: pre-toolbar time before the toolbar appeared (or the cursor warped for WC), time spent in the toolbar, the time between correct colour selection and task completion, and overall task completion time. Note that time between successive trials was recorded, but not used in the analysis because participants were told they could rest between each trial. The start time for each trial was when participants first targeted the white square (when they heard the auditory signal). Each trial ended when participants clicked inside the white square after selecting the correct colour from the toolbar. The four time measures are defined as:

- **Completion time**: time from first targeting of the white square to the completion of trial.
- **Pre-toolbar time**: time from first targeting of the white square to the mode switch key being depressed (note that this measure does not exist for Static Toolbar)
- **Time in toolbar**: time from the mode switch key being depressed to the selection of the correct colour from the toolbar
- **Time after selection**: time from the correct colour selection to the task completion.

I also collected data about how far participants moved in each phase of the task, and collected participants' subjective responses in a questionnaire after the study (see Appendix for survey).

4.3.3 System

The study used a custom Objective-C application to present the experimental tasks and record performance data. The four IPI techniques (three in-place techniques plus a static toolbar) were implemented on a Macintosh PC using a custom-programmed iPhone as a trackpad device. The iPhone 'trackpad' used WiFi and a third party library called AsyncSocket

(http://code.google.com/p/cocoaasyncsocket/) to communicate with the PC. It functioned like a typical laptop trackpad: dragging a finger along the surface of the iPhone moved the system cursor on screen, and clicking was accomplished with a tap on the trackpad's surface.

The control to display (C/D) ratio was chosen from pilot studies. It used a step wise constant C/D function. This function first calculates the speed the finger is moving, calculated using the distance travelled and the time elapsed. This speed is then mapped to a factor (using the stepwise function), and the distance travelled is multiplied by the factor to get the final location of the cursor. The stepwise constant function values are shown in Table 2.

Table 2 - Trackpad stepwise constant C/D function

Speed	Factor
0 – 1	1.0
1 – 1.5	1.5
1.5 – 3	2.0
3 – 7	3.0
7 +	3.5

The trackpad was fixed in place in front of a wireless keyboard, and the study system was displayed on a 22-inch LCD monitor with 1600x1200 resolution (see Figure 25). The mode switch to activate the toolbar was implemented as a keyboard shortcut (the 'z' key). The application window was 1280x700 pixels with a 280-pixel sidebar in which the static toolbar appeared. The on-screen toolbars were the same physical size as the toolbar on the trackpad (3.5 inches wide).

Note that all of the conditions used the same trackpad setup. Although Shadow Cursor and Warp Cursor could easily be implemented using a regular mouse, I required all techniques to use the trackpad to control for any experimental noise that could be introduced by changing input devices between conditions.

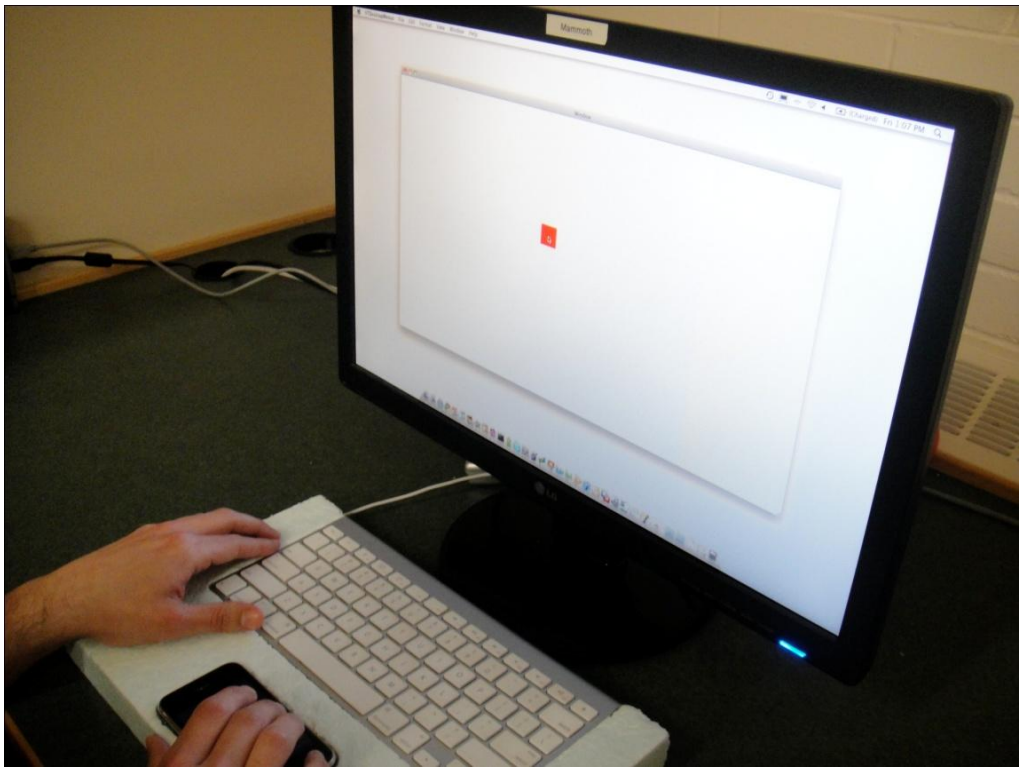


Figure 25 - Experimental setup for study one

4.3.4 Results

We used RM-ANOVAs to test the effects of *toolbar location* on completion time, toolbar time, pre-toolbar time, and path length. Pairwise comparisons were tested using the Least Significant Difference at α=0.05. Outlier trials (where completion time was more than 3 standard deviations above the mean - 36 trials, 0.9%) were removed from further analyses, while trials with selection errors were included.

Completion time

There was a significant main effect of toolbar technique on completion time ($F_{3,45}$=61.7, p≈0.000); see Figure 26. Pairwise comparisons showed that Static Toolbar was significantly slower than all of the in-place toolbars and that SC was significantly faster than WC. There was no significant difference between SC and VT.

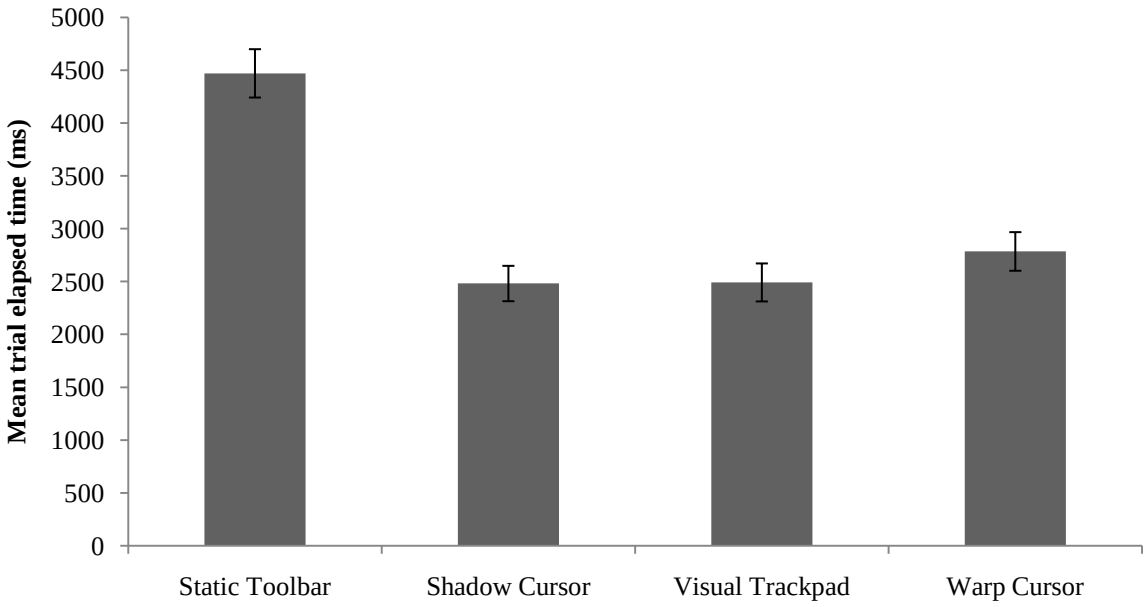


Figure 26 - Mean trial elapsed time in milliseconds

When I subdivided the mean completion time into its sub-parts, I found there were also significant effects within each of the sub-parts. This discussion follows.

Toolbar time

There was a significant main effect on time spent in the toolbar, defined as the time from the mode switch key being depressed to the correct toolbar colour selection ($F_{2,30}$=22.1, p≈0.000); see Figure 27. Pairwise comparisons showed that the three techniques were significantly different; less time was spent in the VT toolbar than in SC or WC; less time was spent in SC than in WC.

Pre-toolbar time

There was a significant main effect on time spent between the first target acquisition and the mode switch ($F_{2,30}$=4.07, p=0.027), defined as the pre-toolbar time. Note that ST was not included in this analysis as it does not have a mode switch. Pairwise comparisons showed that less time was spent in SC than in VT (see Figure 27). There was no significant difference between SC and WC.

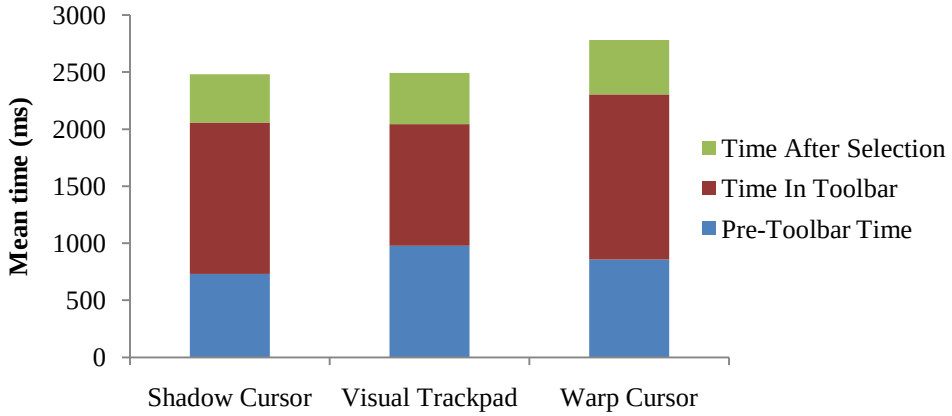


Figure 27 - Breakdown of trial completion time into parts

Path length

There was a significant main effect of toolbar technique on the Euclidean distance travelled by the cursor or the finger, between the mode switch and where users made the colour selection ($F_{2,30}$=149.2, p≈0.000). ST was not included in this analysis as it does not have a mode switch. Pairwise comparisons showed that people moved less with SC than VT or WC (see Figure 28). Note also that Figure 28 shows the path length in pixels, which is the measure of distance for on-screen cursors, but distance for VT depends on DPI (number of pixels per inch). The on-screen toolbars were the same physical size as the VT toolbar (3.5 inches), with on-screen toolbars 280x140 pixels and VT toolbars 480x240 (the max size on the trackpad). The distance presented for VT in Figure 28 has been scaled to match the scale of the on-screen techniques.

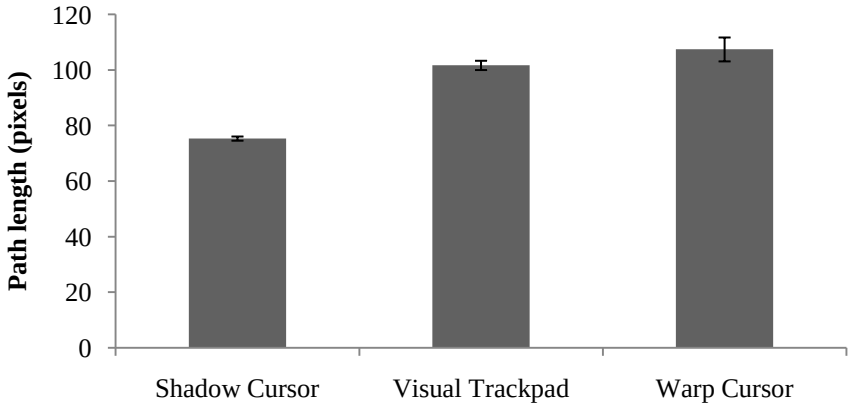


Figure 28 - Mean path length in pixels to select colour

Subjective rankings

Mean rankings of the techniques are shown in Figure 29. A Friedman test (χ^2_3=12.3, p=.006) revealed significant differences between the techniques and Wilcoxon signed ranks tests showed that ST was ranked lower than SC (Z=2.1, p=.033) or VT (Z=2.7, p=.007), but not WC (Z=1.8, p=.080). There were no differences between the three in-place techniques.

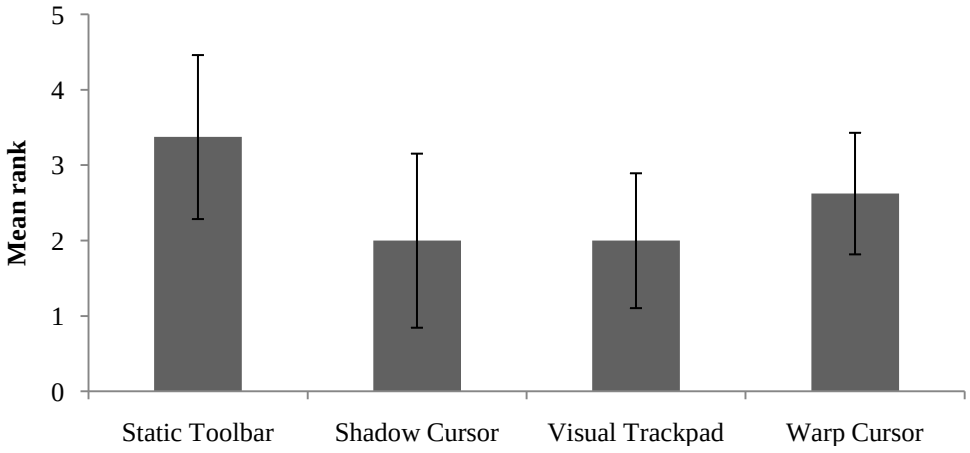


Figure 29 - Mean rank of the four techniques tested (lower is better)

4.4 DISCUSSION

In this section, I first answer my three research questions posed in Section 4.1. I then discuss questions raised from the study, comments from participants, and conclude with a discussion of the limitations of the study.

4.4.1 Answers to the research questions

Previous researchers of attention used response time (the time between the stimulus and when the participants respond) as a measure of attentional shifts [70]. For this study, I also use time as a measure of attentional shifts. Since the task does not require a large cognitive load, the differences between the participants' performance are an indication of the relative attentional costs of the techniques.

What are the attentional costs associated with each of the three conceptual locations in the taxonomy?

First, from Figure 27, participants took significantly longer to activate the toolbar for the outside workspace (VT) technique compared to the in-place (SC) technique. This suggests that users may have been shifting their attention down to the external input device while they were preparing to do the mode switch.

Second, from Figure 26, participants took significantly longer for the inside workspace (WC) technique than for the in-place (SC) technique. This follows my predictions that in-place techniques would be faster than inside workspace techniques; however, there was no significant difference between the in-place (SC) technique and the outside workspace (VT) technique. This is surprising, because I had expected that tool containers displayed outside the workspace would be the slowest.

What is the effect of access method on attention?

The three access methods are in-place popup, on-screen cursor warping, and off-screen direct touch. The results showed that the cursor warping method was significantly slower than the other access methods. In addition, participants moved the cursor farther into the target colour with the warping access method than with the in-place access method. With these two findings, it seems that cursor warping may not be a good access method.

There was no significant difference between the in-place popup and off-screen direct touch access methods. Because the in-place technique is an optimal technique (minimal required cursor movement, and little added attentional costs), this suggests that direct touch is likely a good access method for toolbars.

What are the attentional costs of switching between indirect and direct input?

It seems that the attentional costs of switching between indirect and direct input are negligible compared to the attentional costs for the entire task. Because there was no significant difference

between SC and VT, any additional attentional costs associated with switching between indirect and direct input are offset by the benefits of providing direct input (see Figure 27).

Comments from the participants seem to agree with this. Of the 16 participants, 12 participants answered “agree” or “strongly agree” to the statement “Visual Trackpad was easy to use”. Also, 12 of the 16 participants also answered “agree” or “strongly agree” to the statement “I was able to learn Visual Trackpad quickly”. This suggests that the attentional shift required to switch between indirect and direct is negligible.

Nearby on secondary display or distant on the same display?

In Chapter One, I posed a simple question: “...would it be better to have a nearby tool container on a secondary display, or a more distant tool container on the same display?” I can answer this question by looking at Warp Cursor and Visual Trackpad. Warp Cursor is a distant toolbar on the same display, and Visual Trackpad is a nearby toolbar on a secondary display (nearby because the point of interaction is already in place). The results of the study show that, in fact, a tool container displayed on a nearby secondary display can allow faster selections than more distant tool containers on the same display.

Note that both Warp Cursor and Visual Trackpad afford in-place tool selections, so although the results show that a nearer tool container on a secondary display can be faster than a farther tool container on the same display, this difference includes only the time required to perform the attentional shifts. Although not explicitly investigated in this thesis (but is a topic for future work), another situation is that of a tool container displayed on a secondary display that does not accept direct input. In typical dual-display setups, users often place floating palettes and windows on the secondary display, thereby freeing up the main-screen’s real estate for work artifacts. For this situation, I pose the question again: is it better on the secondary display or on the same display? Fitts’s Law states that the closer an item is, the faster it can be targeted. However, as can be seen from the discussion of the taxonomy of location in Chapter Three, there is more to location than just the distance to target an item.

User performance in dual-display setups depends on the complexity of the workspace on the secondary display, as shown in Figure 30. The more complex the workspace (Figure 30b), the more costly the visual search for the tool container.

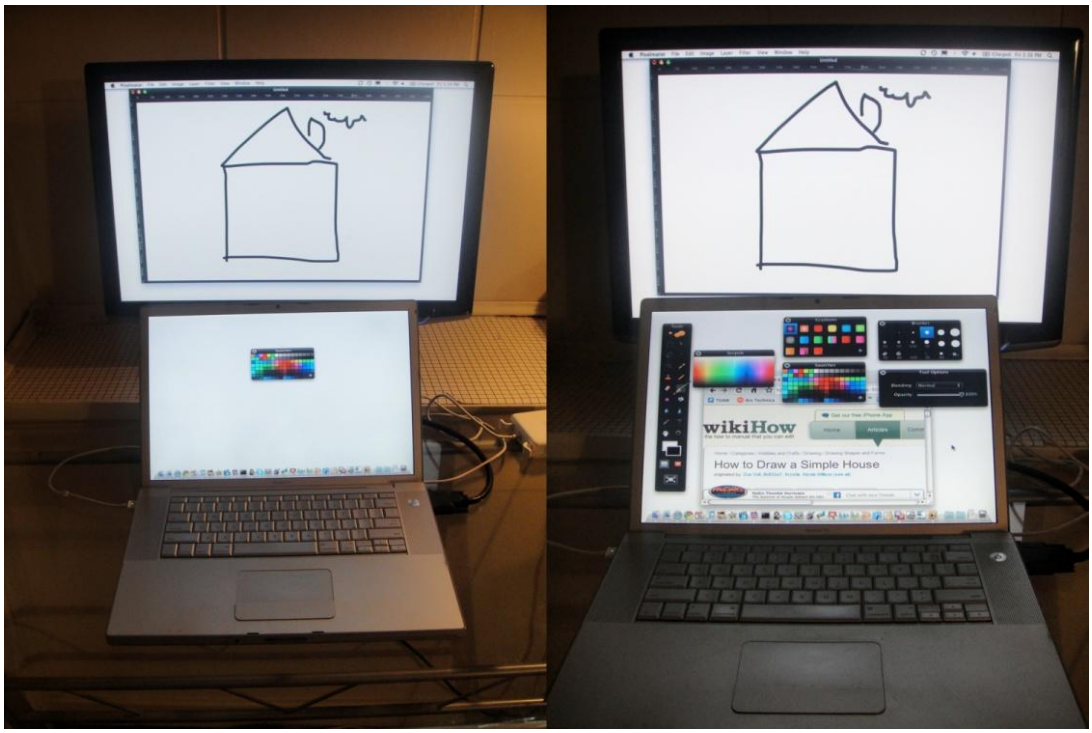


Figure 30 - a) uncluttered secondary display, b) cluttered secondary display

Although there are other factors (like clutter) that can mitigate this, a nearby tool container on a secondary display would be better than a far away tool container on the same display.

4.4.2 Conclusions from the study

There are six main conclusions from this study.

First, the poor performance of WC in Figure 26 suggests that warping may not be a good strategy. One possible reason for WC's poor performance is that the cursor is a small target that is difficult to reacquire, requiring additional search time before users can make a selection from the toolbar. Additional feedback could be added to WC, such as a ripple effect around the cursor when it appears, which could make the visual search for the cursor easier and faster.

Second, from Figure 28, the path length is significantly longer for WC than SC. This is surprising because, after the mode switch, both SC and WC have the cursor displayed at the center of the toolbar. This means that both require the same physical action to select the colours from the toolbars. I suspect that the warping action of the cursor in WC caused participants to move their cursor in a more ballistic manner as compared to SC, thereby causing them to move their cursor farther into the target colour than when selecting from an in-place toolbar.

Third, also from Figure 28, the path length for VT was not significantly different from the path length of WC. This is surprising because participants are shifting their visual attention even farther for VT than they are for WC, and so should create even larger ballistic movements; however, there is one important point to discuss for VT's path length.

The path length for VT is calculated from the location the participant's finger left the trackpad to the point where they selected the colour from the toolbar (scaled appropriately in Figure 28). Because a trackpad is a relative input device, a user's finger could be anywhere on the trackpad when their targeting motion completes (which is to say, when they finish moving their cursor into the white square). Despite this fact, the results shown in Figure 28 suggest that the participants' finger was, on average, near the center of the trackpad at the end of the targeting motion. One possible reason for this could be that users clutch more often than required, resulting in their finger always being closer to the center of the trackpad than the edges. Another, more unlikely, explanation could be that users were simply always closer to the correct colour's location on the trackpad's touchscreen.

Fourth, from Figure 27, participants took significantly longer to do the mode switch for VT than for SC, even though both of these actions are identical. Because VT requires a large visual attentional shift and SC does not, this suggests that users were likely performing the attentional shift before performing the mode switch. This was also observed during the study. In the beginning, participants would perform the mode switch, and then shift their visual attention down to the trackpad. With practice, however, participants began parallelizing the visual shift and mode switch.

Fifth, IPI selections have two benefits. First, users don't have to reacquire the workspace target again. Second, users can parallelize the toolbar selection and the workspace action, performing a double click.

Sixth, and possible the most interesting, the outside workspace technique (VT) was not significantly different from the in-place technique (SC) in mean completion time (see Figure 26). This is likely due to the fact that the outside workspace technique used direct input, which appears to be a good access method for toolbars.

4.4.3 Limitations of the three techniques

The three in-place interaction techniques have some limitations. First, SC obscures the workspace around the point of interaction, as is common for many popup techniques. Participants commented that they often forgot which colour they wanted to select from the toolbar because the toolbar was blocking the target. This could be mitigated with the use of transparency (such as the Hotbox technique); however, because this was a colour matching task, the transparency may cause confusion about which colour to select (either by blending the target colour with the transparent toolbar colour above it, or because the transparency makes the colours look different). Another way of mitigating the occlusion could be to offset the toolbar away from the point of interaction. This would allow users to see the toolbar and the target at the same time, but this would require a larger visual shift (albeit likely not substantial enough to be significant) than the in-place popup.

Second, WC requires visually reacquiring the cursor, which may be difficult for users when the cursor is far from their point of focus; however, because the cursor always appears at the same location on the screen (in the center of the toolbar), this can be mitigated with the use of a ripples effect. The ripples effect would have a much higher attentional draw then the appearance of a small cursor. Also, of the in-place interaction techniques, only WC requires static screen real estate.

Third, VT requires relatively small tool containers because direct touch requires large targets and the trackpad has limited display space; however, this affects all in-place strategies (including

techniques such as Toolglasses [10]). There are several design strategies (such as favourites toolbars, or hierarchical tool containers) that make such an approach feasible. Also, the user's finger will occlude some portion of the trackpad's touchscreen, so users may be required to move their hand completely off of the trackpad and then back again before making a selection.

Fourth, VT requires a second display. In colour-matching tasks, such as the one used in the study, comparing two colours on different displays is difficult due to the colour gamut of each of the displays. In this study, the colours were chosen to be maximally different, and so was likely not an issue. However, this becomes more of a problem when the colour matching must be precise and when comparing colours that are closer. This could be overcome using ICC profiles, and is left as future work.

4.5 CONTRIBUTIONS FROM THIS CHAPTER

There are three contributions from this chapter

First, I provide empirical results showing the effect of tool container location on attention. These empirical results provide a new understanding of the effect of tool container location.

Second, I compared three in-place toolbar access methods, and provide empirical evidence of the subtleties of access method choice.

Third, I developed and tested a novel input device and technique, Visual Trackpad, showing that it provides fast in-place tool selections, while not being as expensive as previously thought.

CHAPTER 5

TOOL CONTAINER LOCATION AND GROUP AWARENESS

The effect of tool container location on individual attention was the focus of Chapter Four. In this chapter, I shift my focus to group awareness, the second important performance factor affected by tool container location that I investigate in this thesis.

Awareness is the "understanding of the activities of others, which provides a context for your own activity" [16]. In multi-user applications, previous research has shown that group awareness is an important factor in supporting effective group collaboration [32,35,45]; however, it is difficult to measure awareness directly [37]. Previous researchers often measure aspects of collaborative work that suggest a lack of awareness; for example, errors and collisions are common ways of indicating a breakdown of awareness [33,37].

One important collaborative working environment is a group of people working around a tabletop. For these applications, territoriality defines how users partition the available space. The location of an artifact in the workspace thus defines such aspects as ownership of the artifact (see Section 3.2), or the function performed on the artifact [18]. When the artifacts are the tool containers, selections made from these tool containers are important for others to be aware of in order to maintain an effective level of group awareness; however, designers still do not fully understand how group awareness is affected by tool container location.

In addition, it is well known that designers are faced with a tradeoff between providing powerful tools for individuals and powerful tools for the group [31]. Because humans have only a limited amount of attention to devote to work they are performing (see attention in Section 2.3), the attentional demands of a user's individual task will affect the amount of attention they can provide to supporting the group work (for example, for awareness). This outlines a tradeoff between attending to individual work and attending to the group task; however, designers still do not fully understand how this tradeoff affects ground awareness.

To investigate these issues, I developed a collaborative tabletop counting game. The game requires players to count to collaboratively by selecting numbers sequentially from toolbars displayed at one of three locations: in the public space, in the users' personal workspaces, and user defined locations (using floating toolbars). With this game, I ran an experiment, using out-of-order selections as indicators of lack of awareness, and the difficulty of a tracking task as a measure of individual task attentional demands.

5.1 RESEARCH QUESTIONS

There are two research questions I want to answer in this chapter.

What is the effect of tool container location on group awareness?

As described in Chapter Three, the location of a tool container can indicate to users properties of the tool container; for example, who owns the container (and thus who can use it), and the association of the tool container's tools to the workspace artifacts. There is, however, one aspect of tool container location that has not been investigated before: its effect on group awareness.

Group awareness has been shown to be important for effective collaboration, and thus it is important for researchers to understand aspects of application design that affect group awareness. One aspect of application design could substantially affect group awareness: tool container location. Specifically, I expect that tool containers that are displayed in shared locations will provide better awareness than tool containers that are displayed in the users' personal workspaces, even though this might not be the optimal location for individual work.

How do the attentional demands of the individual task affect group awareness?

Users have only a limited amount of attention resources. Since collaborative tasks often require users to switch between working individually and working as a group, there is a tradeoff between the attention a user can expend to each of those tasks. Mitigating this tradeoff should be an important design consideration for groupware designers [31].

Although this tradeoff makes intuitive sense, it is important to know how this tradeoff affects individual and group work. I am interested in investigating whether the attentional costs of the individual task will affect group awareness. I expect that the higher the attentional demands are for the individual task, the less attention users will have to expend to the awareness task, which will result in lower group awareness.

5.2 AWARENESS IN TABLETOP APPLICATIONS

Collocated settings intrinsically provide better awareness than distributed settings, because the physical environment can provide awareness cues that are missing when the environment is purely digital. For example, users can see someone walking over to the printer to pick up a new artifact that will be incorporated into the work process. Also, the physical act of picking up a pair of scissors from the center of the table provides other users with the information that this user is going to cut something. These physical awareness cues are missing from distributed environments, and so many researchers have focused on providing artificial awareness cues to distributed teams [e.g., 30].

Even in collocated settings, though, hidden actions still occur. For example, activating a command via a keyboard shortcut is difficult for other users to detect, because the action itself is not easily detectable. On the other hand, reaching into the center of the table to activate a command will be easier for users to notice, and thus provides better awareness.

One important observation can be taken from the above example. Direct input has been shown to provide better awareness in tabletop applications [33], implying that it is the physical action that is important; however, it is not only the physical action that is important to provide better awareness, but also the location of the physical action. Actions that occur in the shared, center, space of the workspace are more likely to be noticed than actions that occur in the personal workspaces of the users. This is possibly due to the fact that it is difficult for users to distinguish a user's regular individual work from actions that affect the group task and are thus important for other users to attend to. Therefore, the location of an action can provide better or worse

awareness. Since many actions that users must be aware of are tool selections, it is thus important to know the effect of tool container location on group awareness.

5.2.1 How to measure awareness

Past researchers have attempted to measure awareness with completion time and errors [33,37,49]. In a collaborative application, an error can be thought of more generally as break downs in awareness. For example, Hornecker et al. [37] used collisions in a collaborative application as a measure of a breakdown of awareness.

Other researchers use a freeze method to measure awareness [17,52]. In this case, the current task is frozen temporarily and the workspace is blanked, and the users are polled about the state of the system. This is likely the easiest way to correctly measure awareness, because users can directly report what they believe to be the current state of the system.

5.2.2 The tradeoff between individual work and group work

Another important observation is that users often switch between working as a group and working individually [31]. It is thus important for groupware designers to provide tools that support both a user's individual work, as well as the group work. Because users have only a limited amount of attention to provide to tasks, there is thus a tradeoff between the amount of attention a user has to expend to their individual work and to supporting the group task (i.e., to stay aware of other users' actions); however, we still do not know the effects of individual task attentional demands on group awareness.

5.3 THE STUDY

To answer the two research questions posed in Section 5.1, I implemented a collaborative tabletop game and ran an empirical study. In this section, I first describe 1-to-N, the collaborative tabletop counting game, and outline the methodology used for the study. I then report the results from this study, which will guide the discussion in the following section.

Note that this study differs from study 1 in one important aspect. There is only one access method (direct touch) in this study, and so each location has only one factor (instead of the interaction between location and access method, as described in Chapter Four).

5.3.1 1-to-N – A collaborative counting game

Overview

1-to-N is a collaborative tabletop counting game. Four players are seated around a tabletop display, each with a direct touch stylus. Each player has their own colour-coded toolbar on the table. The objective of the game is for the players to cooperatively count to N by selecting the numbers sequentially from the toolbars. In addition to collaboratively counting, players must also track a target in their personal workspace. Figure 31 shows four players in a 1-to-N game.

Figure 31 - Four players playing 1-to-N

Rules

Players receive points by selecting the numbers in the correct order and are penalized for selecting numbers out of order. This is the main awareness task in 1-to-N, where players must be continuously aware of the selections of the other players in order to know which number the table is currently on. Out of order selections indicate a lapse in awareness.

The clock in the center of the table resets after each selection. Players are awarded points for correct selections depending on the time left on the clock, so players receive larger scores by correctly selecting numbers quickly. Letting the clock run out launches a mini-game, and the player that had the correct number receives a penalty.

Each player also has a bonus area displayed in their personal workspace. In the bonus area, there is a bonus box. When players hover inside the bonus box, it slowly charges up. Once fully charged, players receive a large bonus. In addition, when players are not hovering inside their bonus box, they receive a penalty for each second they are not correctly hovering. Hovering inside the bonus box is a proxy for a player's individual task, which must be concurrently performed with the group awareness (counting) task.

The player with the highest score at the end of the game wins.

Mini-games

Out of order selections and missed selections (letting the clock run out) launch a mini-game, where a number pad is displayed in each player's personal workspace allowing them to enter the number they believe was the correct next number.

Players who enter the correct number receive a bonus, and players who enter an incorrect number or enter no number receive a penalty. Since players are penalized the same for not entering a number as they are for entering an incorrect number, players are encouraged to guess.

The mini-games provide a direct measure of group awareness. Mini-games are launched by errors (missed selections and incorrect selections), which are caused by lapses in awareness. Also, the answers in the mini-games provides a measure of each players' awareness.

Errors

There are three types of errors that can occur in 1-to-N. The first is selecting a number out of order. This type of error is an indication that this player thought that the group task (the counting task) was at a different stage than it was.

The second type of error is letting the clock run out. Each number selection must occur within a certain time limit, and letting the clock run out indicates that this player was either not aware that they had the correct number, or this player did not know what the correct next number was.

The final type of error is an incorrect response in the mini-game. Since the mini-game is an implementation of the freeze method discussed earlier, errors in the mini-game are a good indication of the state of the group awareness (in addition to particular players' awareness).

Measuring awareness in 1-to-N

The main task in 1-to-N is the group counting task. We can measure the level of group awareness through the errors that players commit during the game. In this sense, it is important that players make errors or else there would be no way to measure awareness.

Through pilot studies, the timing of the clock was determined in order to be slow enough to allow players time to make their selections, but also fast enough to ensure that errors did occur.

Toolbar locations

In order to answer my research question, it was important to vary the location of the toolbars to find the effects of toolbar location on group awareness. The game displays the players' toolbars in three locations: in their personal workspace (see Figure 32a), in the shared space (see Figure 32b), and a variable location via a floating toolbar. These three locations correspond to the three taxonomy locations described in Chapter Three.

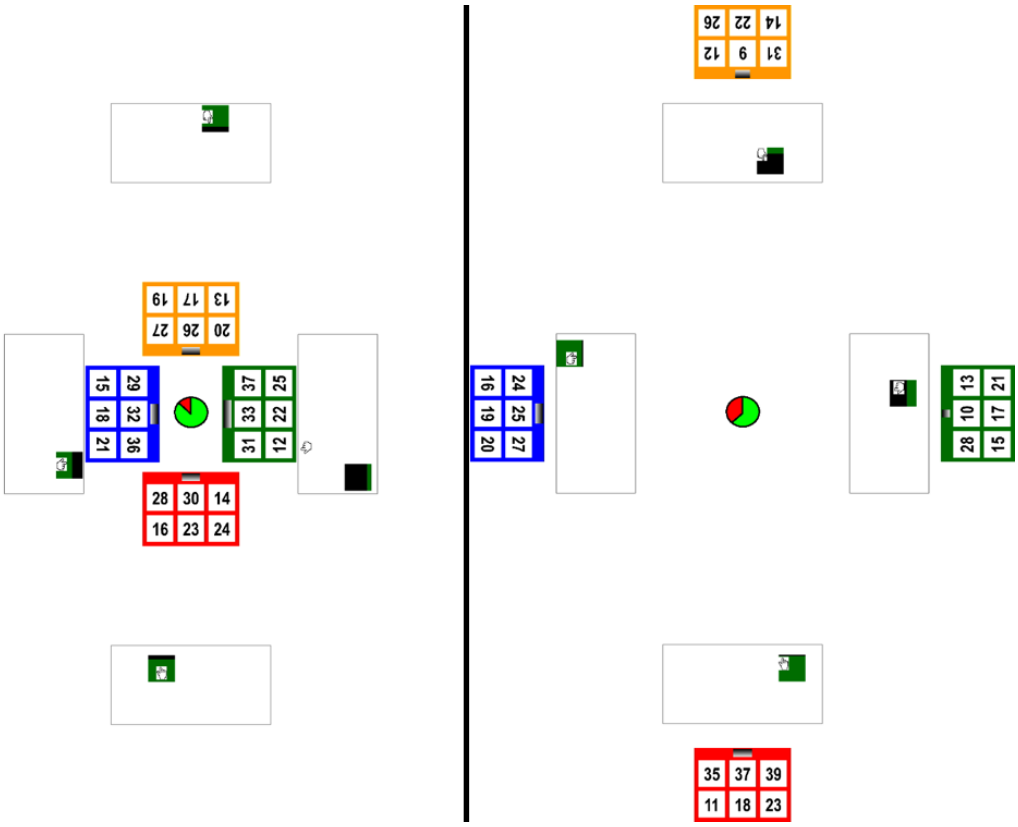


Figure 32 - 1-to-N; a) center location, b) personal location

The floating toolbar corresponds to the in-place location in the taxonomy. Although it is not a popup toolbar, this toolbar can be placed close to the user's personal workspace, and thus corresponds to the in-place location. Note that the floating toolbar can be moved by lifting the pen above a threshold above the table (about 6 inches; see Figure 33b); this causes the toolbar to snap to the user's current point of interaction (it can also be locked in place by tapping on the lock in the upper right corner of the toolbar – see Figure 33a)

Figure 33 - Floating toolbar; a) locked in place, b) being dragged

The second location is along the edges of the table. These locations correspond to the center of each of the users' personal workspaces. This location makes it difficult for other users to see what is in other users' toolbars, especially for the two users on the short ends of the table (red player and yellow player in Figure 32).

The third location for the toolbars is in the center of the table. The center of the table is often the shared space, as in territoriality [60]. In this location, the four toolbars are close together in the center of the table.

Note that in all locations, only one player can make selections from each of the toolbars.

Individual task

In addition to the collaborative counting task, players are also required to attend to an individual task. This individual task is a hovering task, making use of the bonus areas and bonus boxes.

My second research question deals with the tradeoff between attending to the awareness task and attending to individual work. In this study, I vary the attentional demands of the individual task by moving the bonus box at different speeds.

The first speed is not moving, which degrades the hovering task into a docking task. The second and third speed, which I call slow and fast, were chosen from pilot studies. These two speeds require about one glance every 2 seconds and almost constant attention, respectively.

5.3.2 System

The experimental system runs on a large 73" x 49" top-projected tabletop display with a resolution of 1024x1536. Players use custom-built pens as direct touch input to the system. The custom-built pens integrate a digital button into the tip and a Polhemus Liberty (http://www.polhemus.com) sensor attached to the top. The system makes use of a custom-built Polhemus API and the Phidgets API (http://www.phidgets.com), and was written in C#.

5.3.3 Methodology

Participants

Eight participants were recruited from a participant mailing list and were split into two groups of four players. Mean age of participants was 27, 2 were female, and 1 was left-handed. All were experienced computer users (over 30 hours of computer use per week) and none used a touchscreen as their primary pointing device.

Training

Players completed three types of training. First, players were required to count to 10 using the pens and their toolbar in order to make them comfortable with the experimental setup. This was repeated three times: once for each toolbar location. Second, players repeated the counting task in conjunction with taps in a box displayed in their personal workspace. This training introduced players to the bonus boxes and switching their point of focus from inside their workspace to the toolbar. This was repeated three times: once for each toolbar location. Third, players completed three games of 1-to-N to 20, one with each of the toolbar locations. These were shorter versions of the full game. In this training, players were required to make errors, both out of order selections and missed selections, in order to introduce them to the mini-game and the clock.

Data collection

The study used a two factor design: *toolbar location*, with three levels: *center*, *personal*, and *floating*; and *bonus box speed*, with three levels: *not moving*, *slow*, and *fast*.

The same ordering of conditions was used in the testing part of the experiment. Toolbar locations were ordered *center*, *personal*, *floating*. The bonus box speeds were ordered *not moving*, *slow*, and *fast*. Each location was crossed with bonus box speed, for a total of 9 unique conditions.

Each study session consisted of 18 1-to-N games to 40 - two run-throughs of the nine unique conditions. Numbers were randomly distributed among the toolbars, while ensuring that each player received 10 of the numbers to 40.

I collected two main measures: errors (incorrect selections, missed selections, and mini-game errors), and the penalties players received for not correctly hovering in their bonus boxes. As described in Section 5.3.1, errors in 1-to-N are an indication of a breakdown in awareness. Each number selection is recorded, with the timestamp, the selected number, and the correct number. Also, mini-games that were launched by the clock running out (instead of an incorrect selection) were also recorded, and the player that should have made the selection was recorded. I also collected the responses from the mini-games. In addition to these three types of errors, I also recorded the penalties players received from not hovering inside their bonus box. The bonus box penalties provide a measure of how effective players were at their individual task, and the errors provide a measure of how effective players were at the group counting task. Participants also filled out a questionnaire in order to collect their opinions of the system.

5.3.4 Results

Data pre-processing

One participant had a lot of difficulty playing the game. The table location this participant was using had the largest attenuation from the Polhemus tracker, and thus the point of interaction was not the tip of the pen they were holding; instead, the cursor displayed on the tabletop was the point of interaction, and this player did not seem to understand this.

This participant had many more errors than any other participant, and it was obvious when looking at the data that this participant was an outlier. For example, in two games, this participant had zero correct selections. For many other games, over 50% of this participant's selections were errors, compared to the average of 20%. This participant's data was excluded from all analyses.

Errors

There was a main effect of *toolbar location* on selection errors ($F_{2,12}$=10.20, p=0.003, η^2=0.629) Pairwise comparisons showed that players made significantly fewer errors in the *center* condition than the *personal* condition (p=0.021) and the *floating* condition (p=0.004). Also, players made

marginally fewer errors in the *personal* condition than the *floating* condition (p=0.085). Splitting errors into types showed significant effects from incorrect selections ($F_{2,12}$=7.43, p<0.008, η^2=0.553) and marginal effects for missed selections ($F_{2,12}$=3.21, p=0.076, η^2 = 0.349).

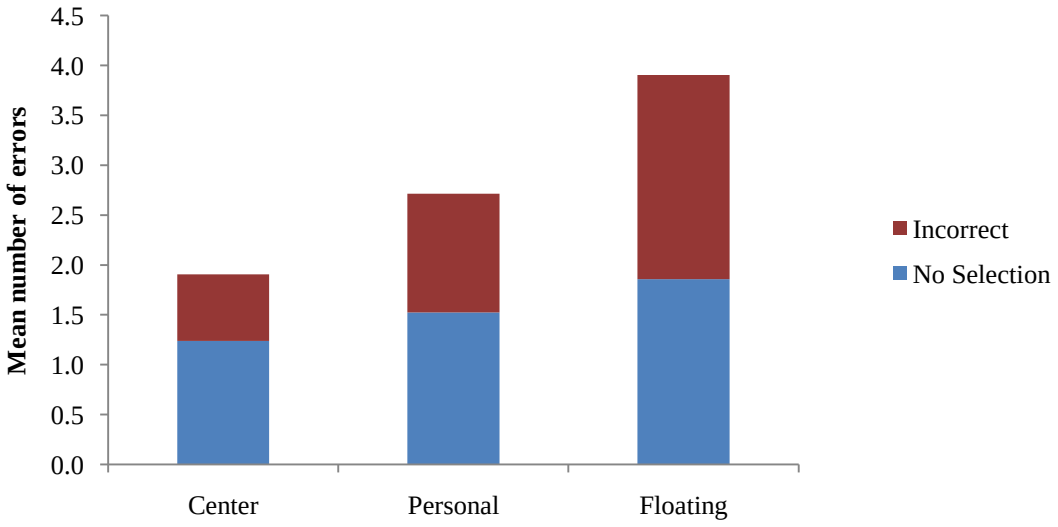


Figure 34 - Mean number of errors per toolbar location

There was no main effect of speed on number of errors ($F_{2,12}$=1.83, p=0.202, η^2=0.234). There was no interaction between speed and location on number of errors ($F_{4,24}$=1.01, p=0.424, η^2=0.144).

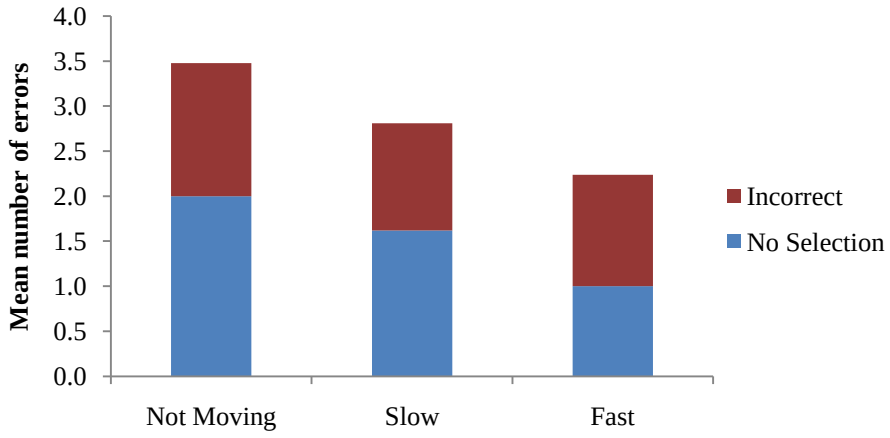


Figure 35 - Mean number of errors per bonus box speed

Mini-game errors

There was a main effect of toolbar location on the number of mini-game errors ($F_{2,12}$=4.00, p=0.047, η^2=0.400). Pairwise comparisons showed that players made significantly fewer mini-game errors in the *center* condition than in the *floating* condition (p=0.039), but there are no significant difference in mini-game errors between the *center* condition and the *personal* condition (p=0.388), nor significant difference in mini-game errors between the *personal* condition and the *floating* condition (p=0.150). Splitting the results into types of errors showed significant effect when the players had the correct number ($F_{2,12}$=6.52, p=0.012, η^2=0.521), and marginal effect when the players did not have the correct number ($F_{2,12}$=3.30, p=0.072, η^2=0.355).

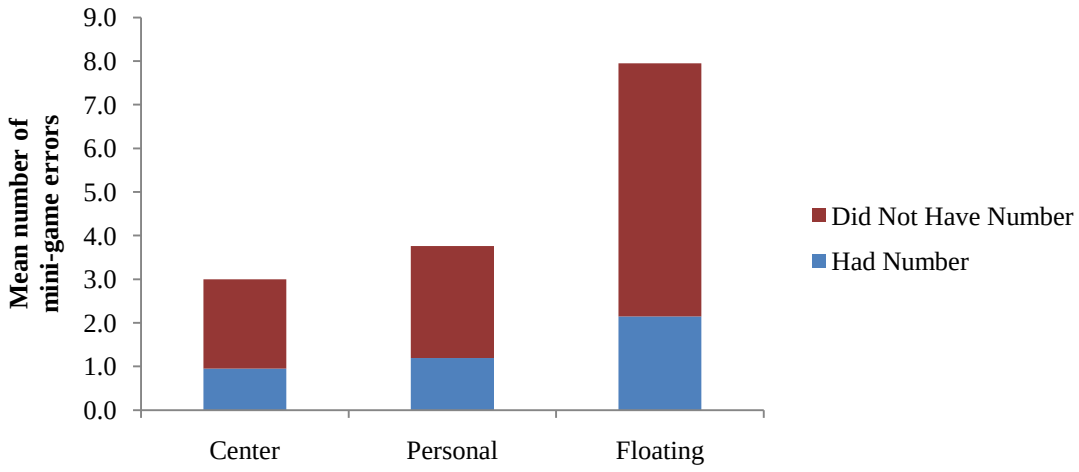


Figure 36 - Mean number of mini-game errors per toolbar location

There was a main effect of speed on mini-game errors ($F_{2,12}$=4.76, p=0.030, η^2=0.442. Pairwise comparison showed players made significantly more errors in the *not moving* condition than the *fast* condition (p=0.024), but no significant difference between the *not moving* condition and the *slow* condition (p=0.151), nor significant difference between the *fast* condition and the *slow* condition (p=0.180). Splitting the results into error types showed there was a significant effect when the players did not have the correct number ($F_{2,12}$=5.78, p=0.017, η^2=0.491), and no significant effect when the players had the correct number ($F_{2,12}$=0.67, p=0.532, η^2=0.100).

There was also a condition by speed interaction on mini-game errors ($F_{4,24}$=3.82, p=0.015, η^2=0.389). For *not moving* and *fast*, there was a difference between *floating* and *center* and *floating* and *personal* (all p<0.050), but *center* was not different than *personal* (all p>0.108). For *slow*, there were no differences between the three locations (all p>0.092). This result is probably not meaningful, and is likely an artifact of only having eight participants.

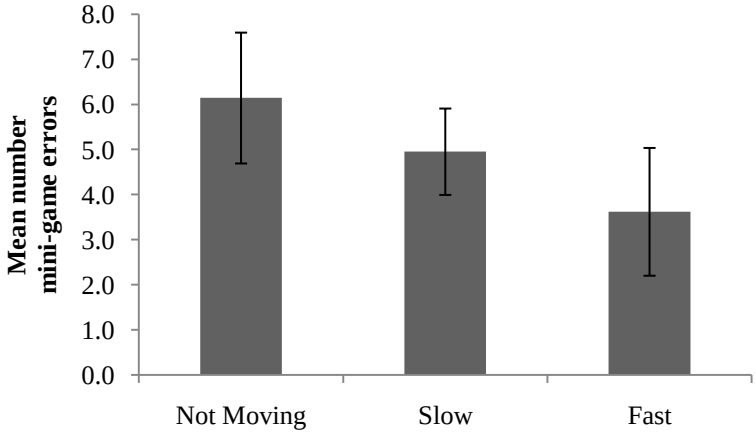


Figure 37 - Mean number of mini-game errors per bonus box speed

Bonus box penalties

There was no main effect of *toolbar location* on the number of *bonus box penalties* ($F_{2,12}$=1.96, p=0.183, η^2=0.246).

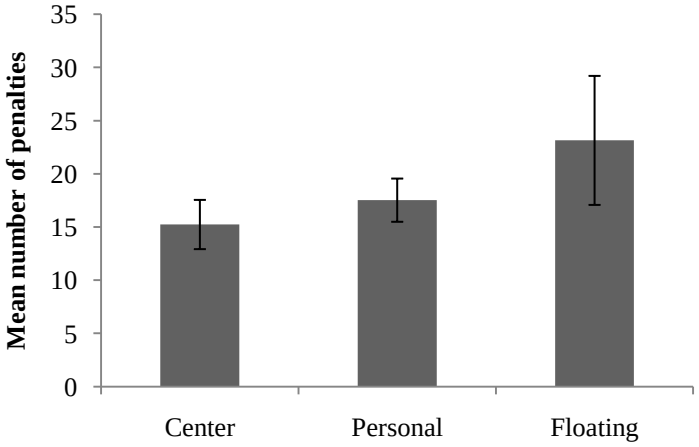


Figure 38 - Mean number of bonux box penalties per location

There was a main effect of *bonus box speed* on the number of *bonus box penalties* ($F_{2,12}$=26.33, p≈0.000, η^2=0.814). Pairwise comparisons showed significant difference between the three conditions: players received significantly fewer penalties in the *not moving* condition than the *slow* condition (p=0.005), significantly fewer penalties in the *not moving* condition than the *fast* condition (p=0.001), and significantly fewer penalties in the *slow* condition than the *fast* condition (p=0.003).

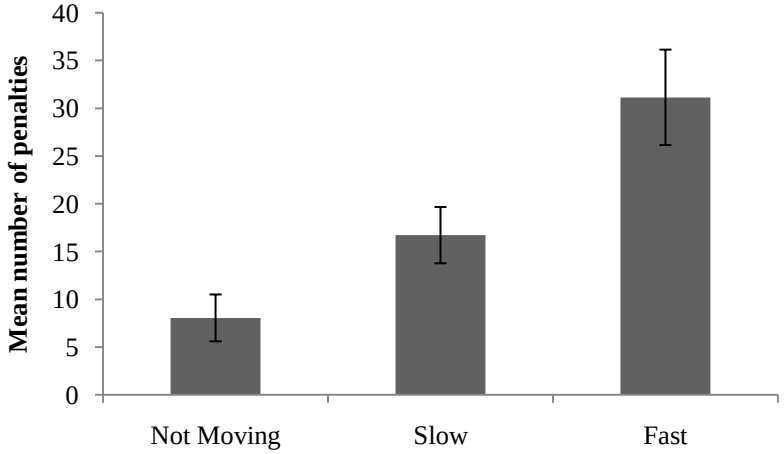


Figure 39 - Mean number of bonus box penalties per bonus box speed

Post-session questionnaire results

When asked to rank the locations based on statements, four players ranked *personal* first and three players ranked *center* first on the statement: "This location made it easy to monitor when other players made a selection".

Four players ranked *personal* first and three players ranked *floating* first for the statement: "This location made it easy to hide my selections from other players". This suggests that the *center* condition did indeed provide better selection awareness, because players could not hide their selections from other players in the *center* condition.

All players answered either "agree" or "strongly agree" to the statement "I was able to adopt strategy that made the game easier" (mean 4.29 out of 5 on Likert scale). A full discussion of player strategies is left for Section 5.4.4.

5.4 DISCUSSION

5.4.1 Answers to the research questions

In this chapter, I posed two research questions I wished to answer with this study.

What is the effect of tool container location on group awareness?

Players made significantly more selection errors when the toolbars were displayed in the personal workspaces than when they were displayed in the shared center space. Similarly, players made significantly more mini-game errors when the toolbars were displayed in the personal workspaces. These results show that tool containers outside of any player's workspace better supports awareness than tool containers inside their workspaces.

How do the attentional demands of the individual task affect group awareness?

As described in the limitations section below, it is difficult to definitively answer this question from the results of this study. There are, however, some interesting conclusions related to this question.

Figure 35 and Figure 37 show that players made *fewer* errors when the individual task becomes more difficult (note that differences in figure 35 were not significant). This seems counter-intuitive; however, Figure 39 shows that players performed *worse* on the individual tracking task as this task became more difficult. Combined, these results show that players would sacrifice their individual performance in order to perform better in the group awareness task.

Because players took the awareness task as the primary task and ignored the individual task in order to perform better on the awareness task, it is difficult to definitively conclude what are the effects of an individual task's demands on group awareness.

5.4.2 The floating condition

From the discussion in Chapter Three, the floating location should have been the optimal location for individual players, because the toolbar can be physically close to the personal

workspace, requiring a smaller visual attention shift than the other locations and faster selections (Fitts's Law); however, from Figures 34 and 36, players made both more selection errors and more mini-game errors in the floating location than the other two locations. The question remains: what happened in the floating condition?

The floating toolbar could be moved around the tabletop display by lifting the pen above a threshold above the table (about six inches). This caused the toolbar to center below the tip of the pen. It was designed in this way to simulate lifting the toolbar and moving it around, and lowering the pen was like placing the toolbar at this location. Unfortunately, players often inadvertently lifted their pens too high, causing the toolbar to center below their pens. Subsequent pen downs were then interpreted as selections from the toolbars, which were often inadvertent, and incorrect, selections.

It was the number of incorrect selections that caused the significant differences between the locations (see Figure 34). This is particularly true for the floating condition, where almost half of the errors were incorrect selections. It is plausible that at least some of these incorrect selections were not awareness errors, but instead inadvertent selections of the wrong number.

Note that the inadvertent movement of the floating toolbar was observed in pilot studies, and so the floating toolbar was modified to include a locking mechanism. The top right of the toolbar housed a lock that when selected would lock the location of the toolbar (see Figure 33). This was meant to avoid the inadvertent movements of the toolbar; unfortunately, the lock was quite small, which players found difficult to select. Also, players often moved their floating toolbars to the edges of the table. The attenuation of the magnetic signal from the Polhemus tracker was larger at the edges of the table, so there was a larger displacement between the pen tip and the point of interaction (cursor) on the table, making the locking mechanism even more difficult.

Lastly, the floating location also caused more overall errors in the mini-game. In this case, players made more errors when they had the correct number than when they did not have the correct number. This suggests that the floating toolbar may have not have supported toolbar item awareness (knowing what is in their toolbar) as well as the other two locations.

5.4.3 Types of awareness in 1-to-N

In general, I am interested in group awareness, specifically selection awareness; however, there are two components to this group awareness. First, players must be aware of which number is next, which is to say, the state of the group counting task. Second, players must be aware of which numbers are in their toolbar, at the least which is their smallest number (and thus their next number to contribute to the group counting task).

Lapses in each of these types of awareness cause different types of errors. Incorrect selections correspond to lapses in the first type of awareness, which is to say, not knowing which number is next. Missed selections (letting the clock run out) correspond to lapses in the second type of awareness, which is to say, not knowing that their smallest number was up next.

The other indicator of lapses in awareness is the mini-game responses. Overall, mini-game responses are a direct indication of the state of awareness in the system, because they are produced using the freeze method; however, I make the distinction between players who incorrectly respond to the mini-game when they had the correct number versus players who incorrectly respond to the mini-game when they did not have the correct number. In the first case, there is a lapse in toolbar item awareness (they did now know what is in their toolbar). In the second case, there is a lapse in table state awareness (they did now know which number came next). Note that these are different, because players often have a large break between selections from their toolbar, and thus may "phase out" for a while, losing track of exactly which number the table is on. As described in Section 5.4.4, one player strategy is to wait for the number just before theirs to be selected. In this case, players may be attending only to the number before theirs, and not attending to all of the selections occurring on the table.

5.4.4 Defining the workspace

In the original version of 1-to-N, there was no individual task; players simply had to count sequentially. During piloting, I found that players would simply hover their cursor over their toolbar and wait for selections to occur. This is not representative of most collaborative work, where users often switch between working individually and working as a group. Also, because

players would simply hover on their toolbars, the two extreme locations (personal at the edges of the table and center in the middle) were used in a similar manner. The only difference between the two was the distance that separates the toolbars, but as seen in Chapter Three, the location of a toolbar means more than just the distance. It was important to define personal and shared areas on the table, such that players would move from their personal workspaces to the shared space.

For these reasons, I introduced the bonus boxes in order to force players to switch from working in their personal workspaces and making selections from the toolbars.

5.4.5 Player strategies

1-to-N is a competitive game, and so players developed different strategies. These strategies could negatively affect the results from the study, so the post-session questionnaire specifically asked players to outline the strategies they adopted.

Five participants stated their most commonly used strategy was to count the number of taps they heard on the table; the other two listed this strategy as the second most used. The next most common strategy was to find the smallest number in their toolbar and search for the number before it on the table; they then simply waited for that number to be selected, thereby not requiring players to be aware of all selections, only the selection just before theirs.

I also asked players which strategy they used to recover when they couldn't remember which number came next. Three said they would enter their smallest number in the mini-games until it was correct, five said they would watch what other players entered in the mini-games, and six said they would watch other players' selections from their toolbars. Only two players indicated that they would launch the mini-game on purpose in order to see what other players would enter.

In both sessions, the most common strategy for finishing the game quickly was for players to announce aloud the numbers they were selecting. During training, both groups began playing the game in this way. This made the game trivial, as players did not have to attend visually to other locations on the screen, suggesting the auditory feedback may be a good way to provide awareness information.

5.4.6 Limitations

Although the study produced interesting results, there are a few limitations to the study.

First, the number of participants is a limitation of this study. The low sample size (of eight participants) likely produced a largely noisy data set due to the large variance between participants. Combined with one removing one participant as an outlier, the results of this study should be considered preliminary until more data confirms the results.

Second, it is difficult to study individual performance in a group study because player performance is inter-reliant. When one player makes an error, it may break the other players' focus because of the mini-game, which could negatively affect their individual task performance as well as their awareness performance. Because of this inter-reliance, it is difficult to isolate individual effects from a group study.

Third, the awareness task in 1-to-N is the primary task and the individual tracking task is the secondary task. This is not as common as the opposite situation, where the awareness task is secondary and subsidiary to the individual work. Regardless, 1-to-N simulates real world tasks that are tightly-coupled, where users must stay constantly aware of the other users' actions, such as air traffic control or subway coordination [35,45].

Fourth, in order to investigate the tradeoff between individual task attentional demands and group awareness, it was important that players expend attentional resources to the individual task. From the results of study 2, it appears that players would ignore the individual task in order to perform better in the awareness task. This was likely due to the fact that the awareness task was the primary task, and players were not forced to perform well in the individual task (except for low scores from penalties).

5.5 CONTRIBUTIONS FROM THIS CHAPTER

There are three contributions from this chapter.

First, I provide empirical results from the study showing the effect of tool container location on awareness. These empirical results provide a new understanding of the effect of tool container location.

Second, I provide an experimental environment where researchers can investigate effects of tool containers properties on group awareness.

Third, I showed that users will expend their attentional resources to the task they believe is primary. In this study, players prioritized the awareness task over the individual task.

CHAPTER 6

GENERAL DISCUSSION

In this chapter, I start with a summary of the findings from the two studies and describe how they affect the taxonomy described in Chapter Three. I then describe some generalizations of the results to other situations and domains.

6.1 SUMMARY OF FINDINGS

In this section, I describe the findings from the two studies and provide context of how these fit into the overall discussion of tool container location.

6.1.1 Tool container location and attention

In Chapter Four, I described a comparison of three in-place interaction techniques. The three in-place interaction techniques were displayed in the three taxonomy locations: in-place, inside workspace, and outside workspace. Each of the taxonomy locations requires different attentional resources in order to make a selection from them. The farther a tool container is from the focus of interaction, the larger the visual shift. Also, there is a visual search cost in order to acquire the tool container before the user can make a tool selection from it.

I found that the costs of these attentional shifts followed intuition (the farther the shift, the more costly it is); however, this cost was offset by the access method, discussed next.

6.1.2 Tool container access method

The second important finding from study 1 is that the access method for in-place interaction techniques has a substantial effect on user performance. Specifically, I looked at three access methods, corresponding to the three in-place interaction techniques: popup, cursor warping, and direct touch on external device.

First, I found that the popup in-place technique was the fastest, but caused occlusion of the workspace around the point of interaction. Second, I found that cursor warping appears not to be a good access method; it performed significantly worse than the other access methods. Third, I found that direct touch on an external input device performed as well as the popup access method, suggesting the direct touch is good for toolbar selections.

6.1.3 Cost of switching between direct and indirect input

Visual Trackpad, described in Chapter Four, is a novel input technique that switches between being an indirect input device and a direct input device. To enable this multiplexing of input type, users must switch between using an indirect input device and using a direct input device. This context switch could be costly for users if it happens often; however, I found that the cost of switching between the two input types appears to be negligible, and that any cost associated with this context switch is outweighed by the benefits of using direct touch.

6.1.4 Tool containers in public spaces better support awareness

In the second study, I compared how tool container location supports awareness. Actions that occur in the public spaces of a shared display are more likely to be noticed by other users compared to actions that occur in their personal workspaces, since it may be difficult to distinguish between actions that affect only the artifacts in the personal workspaces versus actions that effect global artifacts.

Using 1-to-N, a collaborative tabletop counting game, I found that tool containers displayed in the shared space cause less awareness errors than those displayed in the users' personal workspaces, showing that tool containers in the shared space better support group awareness.

6.1.5 Individual task's attentional demands and awareness

Group awareness is important for effective collaboration; however, users must actively attend to the actions of other users in order to stay aware of their actions. A user's attentional resources are limited, and so there is a tradeoff between providing attention to their individual work and providing attention to the group actions.

I expected that the higher the attentional demand of the individual task is, the less users would be aware of other users' actions. There was a main effect of speed on mini-game errors (Figure 37), but it is the opposite of what I would have expected: players made *fewer* errors when the individual task was harder. Note, however, that players performed more poorly on the individual task when the task became harder (Figure 39). Combined, these two results showed that players favored performance on the group awareness task as opposed to on the individual task.

6.1.6 The effects of location

Both study 1 and study 2 investigated the effects of tool container location on important aspects of user performance in two different work contexts.

In the individual context, I found that different tool container locations incur different attentional costs. Specifically, the farther away a tool container is, the more costly the attentional shift; however, the benefit of providing direct input on an external input device appears to offset the additional attentional cost.

In the group context, I found that tool containers outside of each user's workspace better support group awareness as compared to tool containers inside of each user's workspace.

There are few techniques which use the outside workspace location to display tool containers. The results from the two studies presented in this thesis showed that the outside workspace location should be investigated further as a viable alternative to typical tool container locations.

6.2 GENERALIZABILITY

6.2.1 Tool containers

In this thesis, I focused on the effects of tool container location on user performance; however, both studies used one type of tool container: toolbars. There are three main reasons why toolbars are an important tool container to study.

First, toolbars are commonplace in many mainstream applications. Even though they are so common, there are still factors that affect user performance of selections from toolbars that are not well understood. I focused in this thesis on one of these factors: toolbar location.

Second, they are simplest tool containers. Toolbars have flat hierarchies, and thus typical toolbars require simple one-click activations to select items from them.

Third, because they are the simplest, they can be used as placeholders for other types of tool containers. The results from the two studies should transfer to other more complex tool containers, because the investigation of toolbars provides a baseline for how other tool containers are affected by location.

6.2.2 User expertise

Individual work

With practice, muscle memory may allow in-place interaction techniques to perform similarly for all locations because users will not have to shift their visual attention from the workspace. This is particularly possible for direct input, as with Visual Trackpad, since direct touch muscle memory seems more likely than indirect muscle memory.

Group work

A group that works together regularly will likely be able to read their co-workers' actions more easily than groups that do not often work together. Still, with a large workspace like the tabletop display used in study 2, it will be difficult for users to be aware of actions that occur in other users' personal workspaces. Expertise will enable smoother collaborative work, but will likely none the less still have the issues associated with tool container location as described above.

6.2.3 Complex workspaces

In order to isolate the effects of location on individual attention and group awareness, I tested users' performance in simplistic environments. More complex workspaces will likely show the effects of tool container location more than simple workspaces.

For example, in study 1, I found that cursor warping appeared to not be a good access method for in-place interaction toolbars. One observation is that this cursor warping occurs in both Shadow Cursor (the in-place popup technique) and Warp Cursor (on-screen cursor warping technique); however, the number of warps and the distance warped is different for the two techniques. In Shadow Cursor, the cursor warps back to the workspace location the cursor was in before the mode switch, whereas in Warp Cursor, the cursor warps to and from the toolbar. In a more complex workspace, with many work artifacts surrounding the point of interaction, there may be an additional cost to acquire the cursor after the toolbar selection.

Another more complex workspace is an extension of the environment from study 2. In this study, there were two tasks: an individual task and a collaborative group task. In real collaborative work, there are often many more tasks and sub-tasks that occur concurrently with the other tasks. For example, a tabletop display may be a good environment for collaborative document editing. In this case, the group task is to edit the document; however, some individual pages of the document may be edited by only one person, while another subgroup of multiple people are concurrently working on another part of the document. In this situation, not only do users have to stay aware of the collaborative task (edit the document), but there may be other important actions that occur in the individual workspaces of other users (for example, deleting a page).

By studying a simplistic workspace, I can provide a baseline of performance for simple techniques, and assume that more complex workspaces will have additional attentional demands, but will still follow the baseline.

6.2.4 Number of tool container accesses

Although this is highly dependent on the task and application, users make many tool selections from the application's tool containers throughout their work. For this reason, any improvement in tool selection performance can have substantial effects on a user's frustration and overall performance.

As discussed in Section 6.2.2 above (expertise), muscle memory may greatly improve a user's selection speed for direct touch tool containers. This implies that tool items that are accessed

often should be moved to a direct touch input location, as these heavily accessed items will benefit the most from direct touch access.

6.2.5 Beyond tool containers – Notifications

The taxonomy of location could also inform decisions about notification locations. As a simple example, the location of a notification can inform the user of the association of the notification to the workspace artifact or to the application that it is associated (see Section 3.2.3). Notifications that occur outside of a user's workspace should be system-wide notifications, or notifications from other applications, such as new IM messages or when you receive a new e-mail. Notifications that occur inside of a user's workspace, such as the popup messages in Facebook chat, are clearly associated with the current application. Notifications could also appear in-place, such as tool tips that appear above the user's point of interaction when hovering above a workspace item.

6.3 ADVICE FOR INTERFACE DESIGNERS

Based on the results of the two studies and the discussions of their implications, I propose four points of interest for interface designers choosing where to locate their tool containers.

6.3.1 It's not just Fitts's Law

The first, and likely the most important, point is that the selection time (the measure provided by Fitts's Law) is not the only user performance measure affected by tool container location. I have shown in this thesis that other factors of user performance are also affected by tool container location choice, specifically, attention and group awareness. In addition to these user performance factors, I have also described many issues that can arise from tool container properties in the three locations of the taxonomy.

For example, although all tool containers inherently require some screen real-estate, the different tool container locations occlude different parts of the workspace. Fitts's Law states the in-place

tool containers afford faster tool selections than tool containers displayed farther away from the point of interaction; however, in-place tool containers occlude the area of the workspace around the point of interaction, which can cause users to forget which tool they wanted to select (this was observed with Shadow Cursor, in study 1). Hence, although in-place tool containers afford faster selections, there are additional issues introduced by occluding the workspace around the point of interaction.

Another example is that the location of a tool container can inform a user about which item in the workspace it will affect (Section 3.2.4). An in-place tool container provides the most obvious link between the tool and the workspace item, whereas an inside workspace tool container provides at least an application and/or workspace link. Tool containers in the outside workspace location do not necessarily have these intrinsic links, so interface designers should include artificial cues, such as labels or link lines, to inform the user of this link.

6.3.2 Location, location, location

In all direct touch applications, there is occlusion of the tool container by the interaction device (the finger or stylus). This is particularly cumbersome for users on tabletop displays when the tool containers are static, such as the personal location in study 2. In this situation, players often observed that their arm occluded their toolbars, because they had to reach over their toolbars in order to hover in their bonus box (which is to say, interacting in their personal workspace).

This observation is important, because it suggests that for direct input, a toolbar along the edge of the table should not be centered in front of the user. Had the toolbar been offset slightly, players would not have occluded their toolbars with their arms. Another problem players described was that the shadow of their arm occluded their toolbars when they were making selections. This is a problem with all direct touch tool containers.

6.3.3 Outside of workspace may be a good location for tool containers

There are few tool containers that are displayed outside of the user's workspace; however, these tool containers may offer benefits to the users that may not be easily achieved by tool containers

displayed at the other locations. This has been shown to be the case for tabletop displays in study 2, where the shared space better supported group awareness compared to the personal workspaces. It has also been shown to be the case for individual users in study 1, with the good performance of Visual Trackpad.

The taxonomy of tool container location, however, shows there are other potential factors that make the outside workspace location desirable for tool containers. For example, there can be no occlusion of the workspace with outside location workspace, neither by the interaction (the finger, stylus, or embodiment – i.e., cursor) nor by the tool container. Also, outside workspace locations can better support bimanual interactions. The best motivating example of this is keyboard shortcuts, which are efficient once they are learned. Lastly, outside workspace locations can provide different input types and techniques than those offered to the user in the workspace. For example, LensMouse allows users to access toolbar items via direct input, and also supports regular indirect mouse input on screen. This example shows how tool containers should be paired with the best selection technique.

Interface designers should make use of the outside workspace location, because it offers properties that are not always available in the other two locations. Tool containers in this location can also be matched to the best selection technique, allowing users to make faster and more effective selections from them.

6.3.4 Which side of the individual-group tradeoff to support?

As proposed by previous researchers [31], users in collaborative environments must split their attention between attending to their individual work and attending to the group work. As shown in study 2, the location of the tool containers has a substantial effect on the ability for groups to coordinate their work: tool containers displayed in the shared space better support group awareness than tool containers displayed in the users' personal workspaces. These results can be interpreted in two ways.

First, some tool selections are important for all users to be aware of (for example, deleting a page from a shared document). These tools should be displayed in the shared space, because their

selections are more likely to be noticed by other users. This is equivalent to choosing to support the group side of the individual-group tradeoff, because these selections are likely slower for individual users.

Second, some tool selections are either unimportant to other users (for example, using a magnifying glass to see more details) or are tool selections that users want to hide from others (for example, in a game). For these types of selections, designers may wish to specifically reduce the ability of other users to stay aware of these selections by choosing to support the individual side of the individual-group tradeoff. This would mean displaying these tool (and thus their tool containers) inside of the users' personal workspace.

CHAPTER 7

CONCLUSION

7.1 SUMMARY

Interfaces are often organized by grouping tools into tool containers, providing one visual representation for the set of tools. The location of the tool container affects users' performance when selecting from it. Previous researchers have used Fitts's Law, a low-level model of selection performance, to investigate the effects of tool container location on selection time; however, selection time is only one aspect of user performance.

The problem investigated in this thesis is a lack of understanding of the effects of tool container location on two important user performance factors: attention and group awareness. My solution is to provide an initial understanding of the effects of tool container location on attention and group awareness. There were three steps to solving the problem: the development of a taxonomy of tool container location, and two research studies. Through the two studies, I was able to show that tool container location does affect attention and group awareness, and there are many subtleties to individual performance with toolbars.

7.1.1 The taxonomy of tool container location

In order to put the discussion of tool container location in context, I created a taxonomy of tool container location. The taxonomy presents three conceptual locations where tool containers can be displayed: in-place, inside workspace, and outside workspace. It then presents four performance issues – visual shift, visual search, occlusion of the workspace, and association of the tools - and three optional tool container properties – popup, bimanual interaction, and location of interaction. These performance issues and optional properties are affected by the tool container location, and a full discussion is included in Chapter Three.

7.1.2 Research questions

With the taxonomy of tool container location established, I pose five research questions that are answered by the two research studies.

- What are the attentional costs, in terms of visual shifts and visual search, associated with each of the three conceptual locations in the taxonomy?
- What is the effect of access method on attention?
- What are the attentional costs of switching between indirect and direct input?
- What is the effect of tool container location on group awareness?
- How do the attentional requirements of the individual task affect group awareness?

7.1.3 Results from the studies

Fitts's Law states that the farther a tool container is from the point of interaction, the more costly it will be to select from it; however, this appears to be highly dependent on the access method.

I found that the access method of an in-place interaction tool container has a large effect on user performance when selecting from it. First, cursor warping performed significantly worse than the other two in-place interaction techniques, suggesting it is likely not a good solution. Second, in-place popup tool containers are fast but cause occlusion of the important workspace around the point of interaction. The third, and more surprising, result was that direct touch on an external input device appears to be a good solution. This result is two-fold: users did not find shifting their attention to an external input device to be too costly, and switching between direct and indirect input appears to incur little additional attentional costs.

In multi-user applications, the location of a tool container has a substantial effect on the groups' ability to maintain awareness. Toolbars displayed in the shared space better supported group awareness, compared to toolbars displayed in the users' personal spaces. Also, because users in collaborative environments often switch between working alone and working as a group, there is

a tradeoff between the attention a user can provide to the awareness task and the attention a user can provide to their individual work; users will expend attention to the task they believe to be the primary task.

7.2 CONTRIBUTIONS

Main contribution

The main contribution of this thesis is the new understanding of how tool container affects attention (Chapter Four) and group awareness (Chapter Five). This new understanding allowed me to create design recommendations for individual and group contexts (Chapter Six) .

Secondary contributions

There are several secondary contributions of this thesis.

First, I developed a taxonomy of location (Chapter Three) that was used to put my work on tool container location in context. Second, I compared 3 in-place toolbar access techniques (Chapter Four), and provide empirical results of the benefits and drawbacks to the different access techniques. Third, I developed and tested a novel input device and technique (Visual Trackpad, described in Chapter Four). Fourth, I provide an experimental environment where researchers can investigate effects of tool containers properties on group awareness

7.3 FUTURE WORK

The work presented in this thesis provides context for more research into the effects of GUI tool container location on user performance. There are several avenues that could provide more valuable insight into the effects of tool container location.

7.3.1 Eye-tracking

The use of an eye-tracker would provide valuable insight into the visual shifts required to acquire tool containers at various locations. Specifically, I am interested in investigating whether muscle memory of Visual Trackpad's toolbar items allows users to make selections without shifting their visual attention from the workspace. Also, in multi-user environments, users may be able to attend to other users' selections without devoting any visual attention at all (peripheral attention, or auditory attention, such as hearing the users making selections as was seen in study 2).

7.3.2 More complex tool containers

I expect that the results presented in this thesis provide a baseline of user performance for the different locations in the taxonomy. The use of toolbars means that the results from the studies are simply a baseline, and I expect that more complex tool containers would show the same pattern of location-based user performance. However, other factors of more complex tool containers could interact with location. The taxonomy of location in Chapter Three suggests other tool container properties that may interact with location. For example, if the tool container is bimanual, dynamic or static, hierarchical or linear.

7.3.3 Investigate a real world task

In addition to using eye-tracking technology and more complex tool containers, additional studies should be run using a real world task. Some examples of potential tasks have been described in this thesis: collaborative document editing and individual drawing. In addition, most application interfaces have more than one tool container. There may be additional problems introduced when there is more than one tool container; for example, for in-place popup tool containers, users must remember how to activate each of the tool containers, as well as which items are in which tool container. Also, as discussed previously in this thesis, stacked toolbars may require additional visual search time. This extra attentional demand may be mitigated with experience, as users become accustomed to the interface and memorize the locations of the tools they most often use.

7.3.4 Dual-display setup: a common setup missing from study 1

One location that was not investigated in study 1 is tool containers on a secondary monitor; as dual-display setups are becoming commonplace, it is important to also investigate the attentional effects of this location for tool containers.

Typical dual-display environments have both screens close together, and thus conceptually form one continuous screen to the users. In this way, tool containers displayed on a secondary display can be thought of as inside the user's workspace. This provides an example of a tool container displayed inside the user's workspace, but also on another display, which may exhibit some of the user performance issues that are exhibited by outside workspace tool containers.

7.3.5 Artificial awareness cues

Since it is difficult to attend to both individual work and maintain awareness of others' actions, designers should provide artificial awareness cues that require less attentional demands. For example, displaying a notification in a user's personal workspace can provide them with the group awareness they require, and also does not require the user to shift their attention from their current workspace. There are many ways designers could provide artificial awareness cues, with varying degrees of success, and so it is important to investigate effective methods for providing these awareness cues. Lastly, if the cues are visual, there are likely effects of notification location on user performance, as discussed in Section 6.2.5.

7.4 CONCLUSION

Typical Windows, Icons, Menus, and Pointers (WIMP) interfaces are organized by grouping multiple tools into tool containers, providing one visual representation for the tools. Common examples of tool containers include toolbars and menus, as well as more complex tool containers, like Microsoft Office's Ribbon, Toolglasses, and marking menus.

It is difficult to compare user performance between different tool containers because they have different properties (for example, some are popup tool containers and some have a static visual representation); however, one property of tool containers is constant among all tool containers: they must all have a visual representation. For this reason, *tool container location* is a property shared by all tool containers, and thus we can compare tool containers displayed in the same location. The effects of tool container location on user performance have been studied extensively in the past using Fitts's Law, which governs selection time; however, selection time is only one aspect of user performance.

In this thesis, I showed that tool container location affects other aspects of user performance, specifically attention and awareness; however, designers lack an understanding of the effects of tool container location on two important user performance factors: attention and group awareness. My solution is to provide an initial understanding of the effects of tool container location on these factors. In solving this problem, I developed a taxonomy of tool container location, and carried out two research studies. The two research studies investigate tool container location in two contexts: single-user performance with desktop interfaces, and group performance in tabletop interfaces. Through the two studies, I was able to show that tool container location does affect attention and group awareness, and to provide new recommendations for interface designers.

REFERENCES

1. AskTog: A Quiz Designed to Give You Fitts. http://www.asktog.com/columns/022DesignedToGiveFitts.html.
2. Attention - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Attention.
3. Adams, M.J., Tenney, Y.J., and Pew, R.W. Situation awareness and the cognitive management of complex systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society 37*, 1 (1995), 85–104.
4. Ahlström, D., Cockburn, A., Gutwin, C., and Irani, P. Why it's quick to be square: modelling new and existing hierarchical menu designs. *Proceedings of the 28th international conference on Human factors in computing systems*, ACM (2010), 1371-1380.
5. Albinsson, P.A. and Zhai, S. High precision touch screen interaction. *Proceedings of the SIGCHI conference on Human factors in computing systems*, (2003), 105–112.
6. Anderson, J.R. and Matessa, M. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction 12*, 4 (1997), 391–438.
7. Anderson, J.R., Matessa, M., and Lebiere, C. ACT-R: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction 12*, 4 (1997), 439–462.
8. Balakrishnan, R. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *International journal of human-computer studies 61*, 6 (2004), 857–874.
9. Baudel, T., Buxton, W.A., Fitzmaurice, G.W., et al. *Clickaround tool-based graphical interface with two cursors*. Google Patents, 1997.
10. Bier, E.A., Stone, M.C., Pier, K., Buxton, W., and DeRose, T.D. Toolglass and magic lenses: the see-through interface. *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, (1993), 80.
11. Buxton, W. A three-state model of graphical input. *Human-computer interaction-INTERACT*, (1990), 449–456.
12. Callahan, J., Hopkins, D., Weiser, M., and Shneiderman, B. An empirical comparison of pie vs. linear menus. *Proceedings of the SIGCHI conference on Human factors in computing systems*, (1988), 95–100.
13. Card, S.K., Moran, T.P., and Newell, A. The keystroke-level model for user performance time with interactive systems. *Communications of the ACM 23*, 7 (1980), 396–410.
14. Cherry, E.C. Some experiments on the recognition of speech, with one and with two ears. *Journal of the acoustical society of America 25*, 5 (1953), 975–979.
15. Doucette, A., Gutwin, C., and Mandryk, R.L. A Comparison of Techniques for In-place Toolbars. *Proceedings of Graphics interface 2010*.
16. Dourish, P. and Bellotti, V. Awareness and coordination in shared workspaces. *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, (1992), 114.
17. Endsley, M.R. Measurement of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society 37*, 1 (1995), 65–84.
18. Everitt, K., Shen, C., Ryall, K., and Forlines, C. Modal spaces: spatial multiplexing to mediate direct-touch input on large displays. *CHI'05 extended abstracts on Human factors in computing systems*, (2005), 1359–1362.

19. Fisher, D.L., Coury, B.G., Tengs, T.O., and Duffy, S.A. Minimizing the time to search visual displays: The role of highlighting. *Human Factors: The Journal of the Human Factors and Ergonomics Society 31*, 2 (1989), 167–182.
20. Fitts, P.M. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology: General 121*, 3 (1954), 262–269.
21. Fitts, P.M. and Peterson, J.R. Information capacity of discrete motor responses. *Journal of Experimental Psychology 67*, 2 (1964), 103–112.
22. Fitzmaurice, G., Khan, A., Pieké, R., Buxton, B., and Kurtenbach, G. Tracking menus. *Proceedings of the 16th annual ACM symposium on User interface software and technology*, (2003), 71–79.
23. Fitzmaurice, G., Matejka, J., Khan, A., Glueck, M., and Kurtenbach, G. PieCursor: merging pointing and command selection for rapid in-place tool switching. *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, (2008), 1361–1370.
24. Fleetwood, M.D. and Byrne, M.D. Modeling the visual search of displays: a revised ACT-R model of icon search based on eye-tracking data. *Human-Computer Interaction 21*, 2 (2006), 153–197.
25. Forlines, C., Vogel, D., and Balakrishnan, R. HybridPointing: fluid switching between absolute and relative pointing with a direct input device. *Proceedings of the 19th annual ACM symposium on User interface software and technology*, ACM (2006), 211-220.
26. Galitz, W.O. *The essential guide to user interface design: an introduction to GUI design principles and techniques*. John Wiley & Sons, Inc. New York, NY, USA, 2002.
27. Grossman, T. and Balakrishnan, R. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. *Proceedings of the SIGCHI conference on Human factors in computing systems*, (2005), 281–290.
28. Grossman, T., Hinckley, K., Baudisch, P., Agrawala, M., and Balakrishnan, R. Hover widgets: using the tracking state to extend the capabilities of pen-operated devices. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, ACM (2006), 861-870.
29. Guimbretiére, F. and Winograd, T. FlowMenu: combining command, text, and data entry. *Proceedings of the 13th annual ACM symposium on User interface software and technology*, (2000), 213–216.
30. Gutwin, C. and Greenberg, S. Workspace awareness for groupware. *Conference companion on Human factors in computing systems: common ground*, (1996), 208–209.
31. Gutwin, C. and Greenberg, S. Design for individuals, design for groups: tradeoffs between power and workspace awareness. *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, (1998), 216.
32. Gutwin, C. and Greenberg, S. A descriptive framework of workspace awareness for real-time groupware. *Computer Supported Cooperative Work (CSCW) 11*, 3 (2002), 411–446.
33. Ha, V., Inkpen, K.M., Whalen, T., and Mandryk, R.L. Direct intentions: the effects of input devices on collaboration around a tabletop display. (2006).
34. Harslem, E. and Nelson, L.R. A retrospective on the development of Star. *Proceedings of the 6th international conference on Software engineering*, (1982), 377–383.
35. Heath, C. and Luff, P. Collaborative activity and technological design: Task coordination in

London Underground control rooms. *Proceedings of the second conference on European Conference on Computer-Supported Cooperative Work*, (1991), 80.
36. Hinckley, K., Guimbretiere, F., Baudisch, P., Sarin, R., Agrawala, M., and Cutrell, E. The springboard: multiple modes in one spring-loaded control. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, (2006), 190.
37. Hornecker, E., Marshall, P., Dalton, N.S., and Rogers, Y. Collaboration and interference: awareness with mice or touch input. *Proceedings of the ACM 2008 conference on Computer supported cooperative work*, (2008), 167–176.
38. John, B.E. and Kieras, D.E. The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction (TOCHI) 3*, 4 (1996), 320–351.
39. Kabbash, P. and Buxton, W. The" Prince" Technique: Fills' Law and Selection Using Area Cursors. .
40. Kabbash, P., Buxton, W., and Sellen, A. Two-handed input in a compound task. *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, (1994), 417–423.
41. Kurtenbach, G. and Buxton, W. The limits of expert performance using hierarchic marking menus. *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, (1993), 487.
42. Kurtenbach, G. and Buxton, W. User learning and performance with marking menus. *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, (1994), 264.
43. Kurtenbach, G., Fitzmaurice, G.W., Owen, R.N., and Baudel, T. The Hotbox: efficient access to a large number of menu-items. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, (1999), 237.
44. Lane, D.M., Napier, H.A., Peres, S.C., and Sándor, A. Hidden costs of graphical user interfaces: Failure to make the transition from menus and icon toolbars to keyboard shortcuts. *International Journal of Human-Computer Interaction 18*, 2 (2005), 133–144.
45. MacKay, W.E. Is paper safer? The role of paper flight strips in air traffic control. *ACM Transactions on Computer-Human Interaction (TOCHI) 6*, 4 (1999), 340.
46. MacKenzie, I.S. Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human-Computer Interaction 7*, 1 (1992), 91–139.
47. MacKenzie, I.S. and Buxton, W. Extending Fitts' law to two-dimensional tasks. *Proceedings of the SIGCHI conference on Human factors in computing systems*, (1992), 226.
48. Muller, M.J. Multifunctional cursor for direct manipulation user interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (1988), 89-94.
49. Nacenta, M.A., Pinelle, D., Stuckel, D., and Gutwin, C. The effects of interaction technique on coordination in tabletop groupware. *Proceedings of Graphics interface 2007*, (2007), 198.
50. Odell, D.L., Davis, R.C., Smith, A., and Wright, P.K. Toolglasses, marking menus, and hotkeys: a comparison of one and two-handed command selection techniques. *Proceedings of Graphics Interface 2004*, (2004), 17–24.
51. Parker, J.K., Mandryk, R.L., and Inkpen, K.M. TractorBeam: seamless integration of local and remote pointing for tabletop displays. *Proceedings of Graphics interface 2005*, (2005), 40.

52. Pinelle, D., Nacenta, M., Gutwin, C., and Stach, T. The effects of co-present embodiments on awareness and collaboration in tabletop groupware. *Proceedings of graphics interface 2008*, (2008), 1–8.
53. Plumlee, M.D. and Ware, C. Zooming versus multiple window interfaces: Cognitive costs of visual comparisons. *ACM Transactions on Computer-Human Interaction (TOCHI) 13*, 2 (2006), 209.
54. Pook, S., Lecolinet, E., Vaysseix, G., and Barillot, E. Control menus: excecution and control in a single interactor. *CHI'00 extended abstracts on Human factors in computing systems*, (2000), 264.
55. Price, S., Falc\ ao, T.P., Sheridan, J.G., and Roussos, G. The effect of representation location on interaction in a tangible learning environment. *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, (2009), 85–92.
56. Rubio, J.M. and Janecek, P. Floating pie menus: enhancing the functionality of contextual tools. *Proceedings of ACM UIST 2002 Conference Companion*, (2002), 39–40.
57. Salvucci, D.D. An integrated model of eye movements and visual encoding. *Cognitive Systems Research 1*, 4 (2001), 201–220.
58. Schmidt, K. The problem with 'awareness'. *Computer Supported Cooperative Work 11*, 3 (2002), 285–298.
59. Scott, S.D., Grant, K.D., and Mandryk, R.L. System guidelines for co-located, collaborative work on a tabletop display. *Proceedings of the eighth conference on European Conference on Computer Supported Cooperative Work*, (2003), 178.
60. Scott, S.D., Sheelagh, M., Carpendale, T., and Inkpen, K.M. Territoriality in collaborative tabletop workspaces. *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, (2004), 294–303.
61. Sears, A. and Shneiderman, B. High precision touchscreens: design strategies and comparisons with a mouse. *International Journal of Man-Machine Studies 34*, 4 (1991), 593–613.
62. Sears, A. and Shneiderman, B. Split menus: effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction (TOCHI) 1*, 1 (1994), 51.
63. Sellen, A.J., Kurtenbach, G.P., and Buxton, W.A. The prevention of mode errors through sensory feedback. *Human-Computer Interaction 7*, 2 (1992), 141–164.
64. Shen, C., Hancock, M.S., Forlines, C., and Vernier, F.D. CoR2D: Context-Rooted Rotatable Draggables for Tabletop Interaction. *extended abstracts, Int'l Conf. Human Factors in Computing Systems*, 1781–1784.
65. Sugimoto, M., Hosoi, K., and Hashizume, H. Caretta: a system for supporting face-to-face collaboration by integrating personal and shared spaces. *Proceedings of the SIGCHI conference on Human factors in computing systems*, (2004), 48.
66. Tam, J. and Greenberg, S. A framework for asynchronous change awareness in collaborative documents and workspaces. *International Journal of Human-Computer Studies 64*, 7 (2006), 583–598.
67. Toney, A. and Thomas, B.H. Applying reach in direct manipulation user interfaces. *Proceedings of the 18th Australia conference on Computer-Human Interaction: Design: Activities, Artefacts and Environments*, (2006), 396.
68. Treisman, A.M. and Gelade, G. A feature-integration theory of attention. *Cognitive*

psychology 12, 1 (1980), 97–136.
69. Villar, N., Izadi, S., Rosenfeld, D., et al. Mouse 2.0: multi-touch meets the mouse. *Proc. UIST*, 33–42.
70. Wickens, C.D. and McCarley, J.S. *Applied attention theory*. CRC, 2007.
71. Yang, X.D., Mak, E., McCallum, D., Irani, P., Cao, X., and Izadi, S. LensMouse: augmenting the mouse with an interactive touch display. *Proceedings of the 28th international conference on Human factors in computing systems*, (2010), 2431–2440.
72. Zhai, S., Morimoto, C., and Ihde, S. Manual and gaze input cascaded (MAGIC) pointing. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, ACM (1999), 246-253.
73. Zhao, S. and Balakrishnan, R. Simple vs. compound mark hierarchical marking menus. *Proceedings of the 17th annual ACM symposium on User interface software and technology*, (2004), 33–42.

APPENDICES

STUDY CONSENT FORMS AND QUESTIONNAIRES

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF SASKATCHEWAN
INFORMED CONSENT FORM

UNIVERSITY OF SASKATCHEWAN

Research Project: Techniques for in-place toolbar selections

Investigators: Dr. Regan Mandryk, Department of Computer Science (966-4888)
Dr. Carl Gutwin, Department of Computer Science (966-4888)
Andre Doucette, Department of Computer Science

This consent form, a copy of which has been given to you, is only part of the process of informed consent. It should give you the basic idea of what the research is about and what your participation will involve. If you would like more detail about something mentioned here, or information not included here, please ask. Please take the time to read this form carefully and to understand any accompanying information.

This study is concerned with understanding what factors effect in-place toolbar selections. The goal of the research is to determine the most effective technique for in-place toolbar selections using a trackpad.

The session will require approximately 45 to 60 minutes, during which you will be asked to complete several toolbar selections using a novel trackpad.

At the end of the session, you will be given more information about the purpose and goals of the study, and there will be time for you to ask questions about the research.

The data collected from this study will be used in articles for publication in journals and conference proceedings.

As one way of thanking you for your time, we will be pleased to make available to you a summary of the results of this study once they have been compiled (usually within two months). This summary will outline the research and discuss our findings and recommendations. If you would like to receive a copy of this summary, please write down your email address here.

Contact email address:_____

All personal and identifying data will be kept confidential. If explicit consent has been given, textual excerpts, photographs, or video recordings may be used in the dissemination of research results in scholarly journals or at scholarly conferences. Anonymity will be preserved by using pseudonyms in any presentation of textual data in journals or at conferences. The informed consent form and all research data will be kept in a secure location under confidentiality in accordance with University policy for 5 years post publication. Do you have any questions about this aspect of the study?

You are free to withdraw from the study at any time without penalty and without losing any advertised benefits. Withdrawal from the study will not affect your academic status or your access to services at the university. If you withdraw, your data will be deleted from the study and destroyed.

Your continued participation should be as informed as your initial consent, so you should feel free to ask for clarification or new information throughout your participation. If you have further questions concerning matters related to this research, please contact:

- Dr. Regan Mandryk, Assistant Professor, Dept. of Computer Science, (306) 966-4888, regan@cs.usask.ca
- Dr. Carl Gutwin, Professor, Dept. of Computer Science, (306) 966-4888, gutwin@cs.usask.ca

Your signature on this form indicates that you have understood to your satisfaction the information regarding participation in the research project and agree to participate as a participant. In no way does this waive your legal rights nor release the investigators, sponsors, or involved institutions from their legal and professional responsibilities. If you have further questions about this study or your rights as a participant, please contact:

- Dr. Regan Mandryk, Assistant Professor, Dept. of Computer Science, (306) 966-4888, regan@cs.usask.ca
- Dr. Carl Gutwin, Professor, Dept. of Computer Science, (306) 966-4888, gutwin@cs.usask.ca
- Office of Research Services, University of Saskatchewan, (306) 966-4053

Participant's signature:_____

Date:_____

Investigator's signature:_____

Date:_____

A copy of this consent form has been given to you to keep for your records and reference. This research has the ethical approval of the Office of Research Services at the University of Saskatchewan.

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF SASKATCHEWAN
INFORMED CONSENT FORM

UNIVERSITY OF SASKATCHEWAN

Research Project: Selection Location and Awareness on a Tabletop

Investigators: Andre Doucette, Department of Computer Science (andre.doucette@usask.ca, (306) 966-4888)
Dr. Carl Gutwin, Department of Computer Science (gutwin@cs.usask.ca, (306) 966-4888)
Dr. Regan Mandryk, Department of Computer Science (regan@cs.usask.ca, (306) 966-4888)

This consent form, a copy of which has been given to you, is only part of the process of informed consent. It should give you the basic idea of what the research is about and what your participation will involve. If you would like more detail about something mentioned here, or information not included here, please ask. Please take the time to read this form carefully and to understand any accompanying information.

This study is concerned with understanding the effects the location of selections has on awareness in tabletop applications. The goal of the research is to determine the most effective location for tools in tabletop applications.

The session will require approximately 60 to 90 minutes, during which you will be asked to play a tabletop game with 3 other players. In this game, you will be asked to make several toolbar selections using a custom built electronic pen on a tabletop display.

At the end of the session, you will be asked to fill out a short questionnaire. You will then be given more information about the purpose and goals of the study, and there will be time for you to ask questions about the research.

The data collected from this study will be used in articles for publication in journals and conference proceedings.

As one way of thanking you for your time, we will be pleased to make available to you a summary of the results of this study once they have been compiled (usually within two months). This summary will outline the research and discuss our findings and recommendations. If you would like to receive a copy of this summary, please write down your email address here.

Contact email address: _____

All personal and identifying data will be kept confidential. If explicit consent has been given, textual excerpts, photographs, or video recordings may be used in the dissemination of research results in scholarly journals or at scholarly conferences. Anonymity will be preserved by using pseudonyms in any presentation of textual data in journals or at conferences. The informed consent form and all research data will be kept in a secure location under confidentiality in accordance with University policy for 5 years post publication. Do you have any questions about this aspect of the study?

You are free to withdraw from the study at any time without penalty and without losing any advertised benefits. Withdrawal from the study will not affect your academic status or your access to services at the university. If you withdraw, your data will be deleted from the study and destroyed.

Your continued participation should be as informed as your initial consent, so you should feel free to ask for clarification or new information throughout your participation. If you have further questions concerning matters related to this research, please contact:

- Dr. Carl Gutwin, Professor, Dept. of Computer Science, (306) 966-4888, gutwin@cs.usask.ca
- Dr. Regan Mandryk, Assistant Professor, Dept. of Computer Science, (306) 966-4888, regan@cs.usask.ca

Your signature on this form indicates that you have understood to your satisfaction the information regarding participation in the research project and agree to participate as a participant. In no way does this waive your legal rights nor release the investigators, sponsors, or involved institutions from their legal and professional responsibilities. If you have further questions about this study or your rights as a participant, please contact:

- Dr. Carl Gutwin, Professor, Dept. of Computer Science, (306) 966-4888, gutwin@cs.usask.ca
- Dr. Regan Mandryk, Assistant Professor, Dept. of Computer Science, (306) 966-4888, regan@cs.usask.ca
- Office of Research Services, University of Saskatchewan, (306) 966-4053

Participant's signature: _____

Date: _____

Investigator's signature: _____

Date: _____

A copy of this consent form has been given to you to keep for your records and reference. This research has the ethical approval of the Office of Research Services at the University of Saskatchewan.

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF SASKATCHEWAN
VIDEO RECORDING CONSENT FORM

Research Project:	Selection Location and Awareness on a Tabletop
Investigators:	Andre Doucette, Department of Computer Science (966-4888)
	Dr. Carl Gutwin, Department of Computer Science (966-4888)
	Dr. Regan Mandryk, Department of Computer Science (966-4888)

VIDEO RECORDINGS

"I, _____, agree to allow video recordings taken during the experiment to be used for public presentation of the research results in the manner described in the consent form. However, I understand that I will be given the opportunity to view any video recordings that are intended for public participation and to withdraw consent for them to be reported, if so desired. I also understand that I will receive a copy of any video recordings presented publically for my records."

Participant		**Investigator**	
Name:	_____	Name:	_____
Signature:	_____	Signature:	_____
Date:	_____	Date:	_____

Techniques for in-place toolbar selection

Post experiment survey

1.Participant: _____

2.Please rate your agreement with the following statement for each technique:
I liked this technique.

	Disagree Strongly	Disagree	Disagree Slightly	Neutral	Agree Slightly	Agree	Agree Strongly
Shadow Cursor							
Visual Trackpad							
Warp Cursor							
Normal Toolbar							

3.Please rate your agreement with the following statement for each technique: ***This technique was easy to use.***

	Disagree Strongly	Disagree	Disagree Slightly	Neutral	Agree Slightly	Agree	Agree Strongly
Shadow Cursor							
Visual Trackpad							
Warp Cursor							
Normal Toolbar							

4.Please rate your agreement with the following statement for each technique:
I was able to learn this technique quickly.

	Disagree Strongly	Disagree	Disagree Slightly	Neutral	Agree Slightly	Agree	Agree Strongly
Shadow Cursor							
Visual Trackpad							
Warp Cursor							
Normal Toolbar							

5.Please rate your agreement with the following statement for each technique:
I was able to complete the tasks quickly with this technique.

	Disagree Strongly	**Disagree**	**Disagree Slightly**	**Neutral**	**Agree Slightly**	**Agree**	**Agree Strongly**
Shadow Cursor							
Visual Trackpad							
Warp Cursor							
Normal Toolbar							

6.Please rate your agreement with the following statement for each technique:
I was able to complete the tasks accurately with this technique.

	Disagree Strongly	**Disagree**	**Disagree Slightly**	**Neutral**	**Agree Slightly**	**Agree**	**Agree Strongly**
Shadow Cursor							
Visual Trackpad							
Warp Cursor							
Normal Toolbar							

7.Please rate your agreement with the following statement for each technique: ***Shifting my attention between the target and the toolbar was annoying.***

	Disagree Strongly	**Disagree**	**Disagree Slightly**	**Neutral**	**Agree Slightly**	**Agree**	**Agree Strongly**	**N/A**
Shadow Cursor								
Visual Trackpad								
Warp Cursor								
Normal Toolbar								

8.Please rank the following techniques from 1 to 4, 1 being the one you liked the best and 4 being the one you liked the least.

Shadow Cursor	1	2	3	4
Visual Trackpad	1	2	3	4
Warp Cursor	1	2	3	4
Normal Toolbar	1	2	3	4

9.Please feel free to leave any other comments:

Selection Location and Awareness - Questionnaire

1. For each of the following, please circle your response.

   a) Which color was your toolbar (note your pen's cable is the same color):
   Red **Blue** **Orange** **Green**

   b) I am: **Left-handed** **Right-handed**

   Male **Female**

   c) How many hours per week do you typically spend working on a computer:
   1-10 **11-20** **21-30** **31-40** **41+**

   d) What is your primary pointing device when interacting with a computer:
   Mouse **Trackpad** **Trackball** **Touch-screen**

   e) Have you ever used a tabletop display?
   Yes **No**

2. Please enter your age: _____

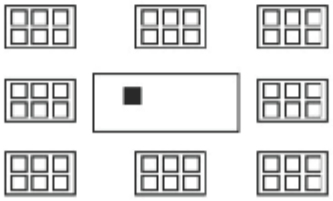
3. For each of the following statements, please **rank each of the toolbar locations** on how well each statement applies to that location by entering a ranking from 1 to 3 (**1 is the best, 3 is the worst**).

	Center	Personal	Floating
This location made it easy to monitor when other players made a selection:			
This location made it easy to keep track of which numbers were in my toolbar:			
This location make it easy to keep track of which numbers other players had:			
This location made it easy to keep my cursor inside my bonus box:			
This location made it easy to hide my selections from other players:			
I was the fastest making my own selections from this location:			

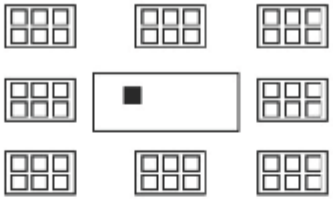
5. For each toolbar location, indicate how difficult the game was with each bonus box speed (**1 - Very easy, 2 - Easy, 3 - Neutral, 4 - Difficult, 5 - Very difficult**):

	Not moving	Slow moving	Fast moving
Center	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5
Floating	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5
Personal space	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5

6. a) Please **circle all** of the locations where you kept your floating toolbar.



b) For each of the floating toolbar locations, rank them from **1-most used** to **8-least used** by annotating the following image (for the locations you didn't use, simply draw an X through that toolbar).



c) Why did you select your most used location (the location you labelled '1' above)? What benefits did it offer to your performance in the game?

_____

_____

_____

_____

7. a) When you did not know which number was next, what strategies did you use to recover (**circle** those which you actively tried in the game):

(A) Watch other players' selections in their toolbars.
(B) Watch other players' selections in the mini-games.
(C) Find my lowest number, and continue to enter the number before my lowest number in the mini-game until it was correct.
(D) Find my lowest number, and find the next lowest on the table. When that number was selected, I knew it was my turn next.
(E) Count the number of taps I could hear other people making.
(F) Count the number of times the clock reset.
(G) Click an incorrect number to launch the mini-game to see what others clicked.
(H) I had a different strategy:

_____

_____

_____

_____

b) Of the strategies you selected above, please rank them from the most commonly used to least commonly used by **placing the strategy's letter** in the corresponding column below:

Rank	1 - Most commonly used	2	3	4	5	6	7	8 - Least commonly used
Strategy								

8. How did you deal with the problem of your arm's shadow occluding your toolbar and bonus box?

_____

_____

_____

_____

9. Players sat on either the long edge (Blue and Green players) or short edge (Red and Orange players) of the table. Did the players in either of these locations (long or short) have an advantage in the game? Which had the advantage? Why was this the case?

_____

_____

_____

_____

10. Please rate the following statements by placing one x per statement:

	Don't know	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
The game was challenging.						
The game was fun.						
The game was easy to learn.						
My arm often obscured my toolbar.						
Other players' arms obscured their toolbars.						
Other players' arms obscured their actions.						
It was difficult to see other players' selections.						
My arm often obscured my bonus box.						
It was difficult to select numbers from my toolbar.						
I was able to adopt a strategy that made the game easier.						
The game was trivial.						

	Don't know	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
I always locked my floating toolbar.						
Within one game, I moved my floating toolbar to more than one location.						
In most games with the floating toolbar, I tried to keep my toolbar at the same location.						
The game was easy to master.						
It was difficult to keep my cursor in my bonus box when it was moving.						
Focusing on my bonus box made it difficult to see when other players made a selection.						
Focusing on other players made it difficult to keep my cursor in my bonus box.						
I watched other players' toolbars more than their actions						
I watched other players' actions instead of their toolbars.						
For most of the game, I knew which number was up next.						
I would like to play this game again.						